Document No 9806-A01-0001

U.S. DEPARTMENT OF DEFENSE

# SMALL BUSINESS INNOVATION RESEARCH PROGRAM TOPIC N96-053 PHASE II FINAL REPORT FOR THE DEVELOPMENT OF VERSATILE HLA INTERFACE UNIT FOR NAVAL AIRCREW TRAINERS

High Level Architecture

TECHNICAL REPORT - STUDY/SERVICES

CDRL SEQUENCE NO. A001

22 December, 1999

Prepared by:

Randall S. Lang

and

Brian Cahill

Principal Investigators

Dual, Incorporated

Simulation & Training Division

30 Skyline Drive

Lake Mary, Florida 32746

| | |
|---|---|
| **Contract Number:** | N00421-98-C-1119 |
| **Contract Value:** | $585,719.00 |
| **Contract Award:** | Competitive |
| **Contract Sponsor:** | Mr. Thomas Drobeck, NAVAIR, PMA-2051BF |
| | Building 2272, UNIT IPT Suite #345 |
| | Patuxent River, MD 20670-1127 |

## TABLE OF CONTENTS

## SECTION I EXECUTIVE SUMMARY

### 1. INTRODUCTION

The purpose of this final report is to present the final products produced and the lessons learned during execution of Naval Air Systems Command Contract N00421-98-C-1119. This project is a Phase II effort of the Small Business Innovation Research (SBIR) Topic N96-053 of Naval Air Systems Command.

There are two independent tasks included in this contract as follows: 1) Tomcat Software Update Mode (TSUM) - Develop a remote software update system for the F-14D trainer; 2) Versatile Flight Trainer Network Interface Unit (VFTNIU) - Develop a High Level Architecture (HLA) compliant interface unit for the F-14D trainer.

The objective of the TSUM task was to develop and implement a process whereby Pt. Mugu personnel can install and test software changes in the Oceana F-14D simulators remotely from the Pt. Mugu TSSA. The primary product of this task is equipment that provides a secure data and voice connection from Pt. Mugu to the F-14D trainer at Oceana. Refer to Appendix A for an overview of the TSUM design.

The objective of the VFTNIU task was to provide a gateway to an HLA network for the F-14D Tomcat trainer. Furthermore this gateway would be designed such that it could be enhanced to support other flight trainers in terms of HLA connectivity. The primary product of this project is a flexible hardware/software package that will initially allow PMA-205 to make the F-14D WST and MFT simulators HLA compliant. This product will be designed to be flexible enough that, with minor modifications, it can be used to make other Navy flight simulators HLA compliant.

### 2. SUMMARY

The week of December 13th Dual delivered and demonstrated software and equipment for this SBIR. See Appendix I for details on the trip. Also see

Appendix B through G for all design and user information associated with this SBIR. The following is a brief summary for each effort.

TSUM – The software and hardware are in place at Oceana to do secure remote data access to the F-14 Trainer as demonstrated the week of December 13th. The maximum buad rate achievable with this equipment is 9600. The stability of this connection can only be determined by continued use. In order to achieve a workable solution, the government must coordinate the reissuing of secure data modems and transfer the remote computer platform to Pt. Mugu. The voice portion of the design is installed but inoperative. At the time of installation, Dual was unable to determine the deficiency. It is recommended that if secure voice is desirable for the TSSA mission that experts in F-14D intercom system design pursue this effort.

VFTNIU – The goal of physical connecting to the F-14D Trainer has been abandoned under this phase of the contract. The physical connection to the F-14D trainer required expensive interface equipment and significant engineering effort that was unable to be funded. The VFTNIU software has been developed and tested with the limitation of no physical connection to the F-14D Trainer. The interface to the F-14D Trainer is terminated in a general purpose shared memory area inside the VFTNIU. Testing of the VFTNIU was achieved by writing emulation software that sends and receives data through the shared memory area.

## 3. CONCLUSION

This SBIR effort has successfully laid the groundwork for continuing efforts on both the TSUM and the VFTNIU. The TSUM effort has supplied the equipment, user's manuals and test of concept for a remote software update capability. In order for this capability to become useable the government must coordinate equipment transfers and implement/adopt security procedures. The VFTNIU effort has supplied a gateway to an HLA network but has fallen short of providing the physical connection necessary for the F-14D to become part of an HLA federation. The equipment, software and documentation provided under this effort is completed such that it could be resumed under another effort.

## SECTION II DETAILED RESULTS

### 1. TOMCAT SOFTWARE UPDATE MODE (TSUM)

### 1.1. PRODUCT DESCRIPTION

The TSUM is a system that provides two secure lines from the F-14D simulator to Pt. Mugu. The data line is a straightforward design consisting of two Windows NT workstations connected by a commercial phone line through two Model 1910 STU III data modems. The work station at the trainer facility connects to the WST Node A host computer through an existing RS-232 connection. Windows terminal emulation is used with existing telecommunications software loaded on the host computer to achieve a virtual terminal at the workstation. Next pcAnywhere is utilized to make this same terminal emulation capability available in Pt. Mugu on the other work station. See Appendix A for a depiction of the data and voice interface. The voice interface requires the manufacturing of a device that converts a telephone signal into one that can be dispatched into the trainers intercom system. This unit design is depicted in Appendix B. See Appendix C and D for the TSUM's detailed design and users guide.

### 1.2. TESTING RESULTS

Dual has completed installation and test of the TSUM. Dual installed and tested the TSUM using building 150 at NAS Oceana as the functional equivalent to Pt. Mugu. The government will be responsible to transition equipment to Pt. Mugu for the final configuration. Only one analog phone line was available at the trainer so this line was used for independent testing of the voice and the data sections. The secure data line was established between building 150 and the trainer using pcAnywhere and the secure data modems (SECTEL Model 1500's). Also, the Intergraph computer system was setup at the TES OPCOM and communication was established with the TES computer.

The voice equipment was installed but was not operational in its final configuration. Communication was established with the Telos Link via phone line but audible communication was not established through the intercom system. The output of the Telos Link feeds into the TES intercom system cabinet. Dual found that troubleshooting of this system was not possible given our limited knowledge and access to this equipment.

## 2. VERSATILE FLIGHT TRAINER NETWORK INTERFACE UNIT (VFTNIU)

### 2.1. PRODUCT DESCRIPTION

The HLA network consists of our Intergraph workstations linked together using Ethernet and the latest version of the Run-Time Interface (RTI) supplied by DMSO. The Stealth viewer from MÄK Technology allows the HLA demonstration to be viewed and the Logger from MÄK allows recording and playback of the HLA demonstration. VRLink from MÄK Technologies supplies tools that allow us to efficiently interface F-14D Tomcat parameters with the HLA network. DUAL developed software is written in Microsoft Visual C++. VRLink's utilities are accessed directly from C++ using a standard API.

A simple F18 simulator program developed by DUAL will be used initially as the HLA participant demonstrating HLA interaction with the F-14D Tomcat simulator. This program implements MÄK's VRLink, simple flight and threat models, and joystick controls to satisfy the demonstration objectives. The DUAL "simulator" is low fidelity but will satisfy the program requirements of demonstrating HLA interactions.

### 2.2. TAP IN POINT FOR THE VFTNIU (LESSONS LEARNED).

The technique used to connect ("tap in") to the F-14D trainer was an essential element for achieving the objectives of this program. Dual was not able to achieve this goal. Dual pursued a HSD approach although other alternatives were feasible. Regardless of the technique chosen, this task requires extensive knowledge/modification of the existing trainer hardware and software. Modifications of this magnitude should be executed under a contract vehicle that is better suited for trainer modifications. The analysis on tap-in point is included here for future consideration.

Determination of Tap In point for the VFTNIU:

DUAL has reevaluated the design of the tap in point for the VFTNIU and presents the following design analysis for review by the government. Upon acceptance by the government, DUAL and the government will work out the implementation details concerned with modification of the Tomcat trainer hardware and software.

Current Design:

Our current design involves using an existing interface to the Tomcat WST (see figure 1). This interface is of type HSD and currently allows one WST to communicate to another WST in integrated mode. Our current design requires disconnecting the HSD line from WST #2 and reconnecting it to our VFTNIU computer. This would allow us to receive and transmit data between the VFTNIU computer and the WST trainer without modifying either hardware or software on the WST side.

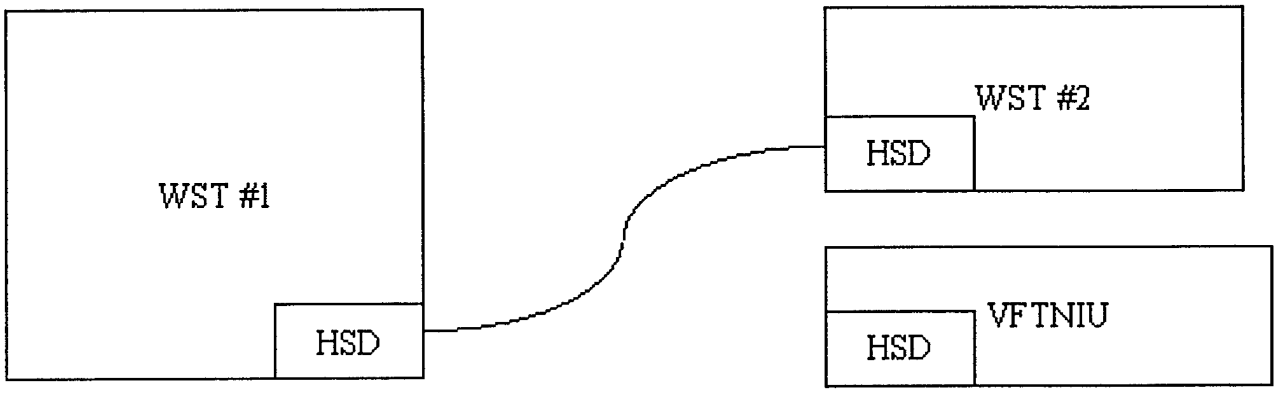


Figure 1. Current Tap In Design

Proposed Design:

The proposed design change involves using an ETHERNET interface from the WST #1 to the VFTNIU (as shown in the figure 2). This involves adding an ETHERNET board to the trainer as well as modification/addition to trainer software.

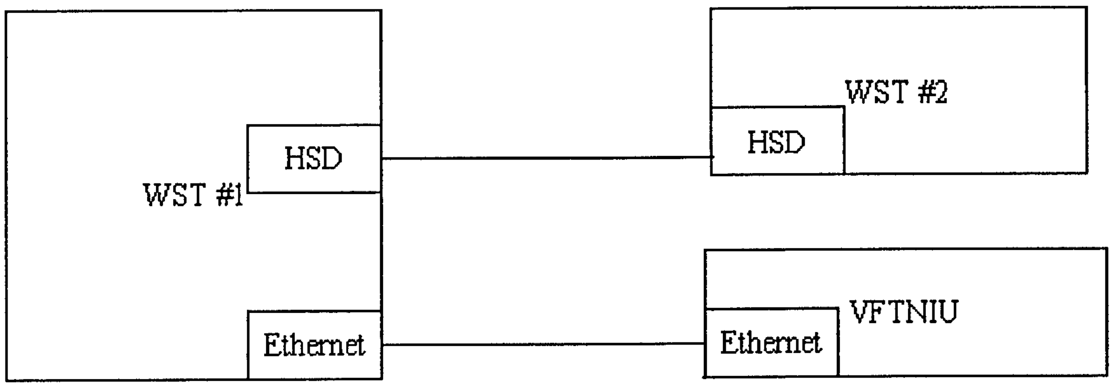


Figure 2. Proposed Tap In Design

Design Change Analysis:

The **advantages** to this design approach are as follows:

- Reduced implementation risk. The HSD to PC interface is an unusual interface. The PC board to implement this is a new product provided from a small company. DUAL's recent experience with a similar design resulted in a late design change. Also functionally emulating the WST #2 adds strict protocol adherence to the interface which may not be able to be attained. The Ethernet interface is of course a common way of interfacing machines of dissimilar architectures.
- Design flexibility and growth potential. The HSD solution defines a static set of data that can't be changed. The data transferred by the Ethernet solution can be modified as requirements change and evolve.

The **disadvantages** are as follows:

- Trainer software must be modified. This includes adding Ethernet driver and a process that loads and extracts required parameters from data pool.
- Trainer Hardware must be modified to include an Ethernet card.

Approach: Dual has decided to pursue the HSD approach. The use of Ethernet or reflective memory is still a valid alternative but would require an indication from the government that this would be a more desirable approach. In addition an alternative approach would require additional support in terms of GFE/GFI. Furthermore, the VFTNIU design encapsulates the specific interface details such that moving to another approach in the future would have minimum impact on the VFTNIU software.

Accomplishments: Under the basic task constraints of schedule and funding, Dual can not accomplish tapping into the F-14D simulator. This task has been moved to the option period where sufficient resources could be applied. The design in place defines the reflective memory block and flexible mapping techniques in the VFTNIU as the interface to/from the VFTNIU. This provides flexibility for future HLA efforts to "tap-in" to the host computer in a number of different ways including HSD.

## 2.3. VFTNIU SOFTWARE DESIGN CONSIDERATIONS

The software design is initiated by the formation of a high level logical model as shown in Appendix G. This model is the culmination of a design effort that involved applying software design considerations to the analysis products. The effort described below will define the logical software design of the VFTNIU system.

The list below encompasses the major design requirements of the VFTNIU effort. While requirement 1 is considered in the analysis effort, the remaining requirements are essential considerations in developing the logical design.

1) Design a system that will satisfy the particular demonstration requirements of this contract. This includes connecting the F-14D trainer to an HLA network and demonstrate aircraft identification/visualization as well as missile tracking/detonation/destruction over the HLA network.
2) Allow for growth of VFTNIU product to include capability to provide HLA interface to other flight simulators. Keep trainer unique logic and dependencies encapsulated
3) Allow for changes in future HLA specifications and products. Keep HLA specific functions and dependencies encapsulated.
4) Minimize cost of implementation and future maintenance and enhancements of the product. Utilize COTS software as much as possible within the above constraints. Consider reuse of other non-proprietary software.

At the heart of the design are two objects identified as HLA GENERATED DATA and HOST GENERATED DATA. These objects provide a neutral interface for storing and accessing HLA and host data. The intent is to define an interface that is HLA friendly but not dependent on HLA specifications or particular product implementations. Well defined methods perform the tasks of storing and accessing data while the mechanisms for database maintenance are encapsulated in the object.

The objects interfacing directly with the HLA network include UPDATE HLA GENERATED DATA and PROCESS HOST GENERATED DATA. These objects contain all references to HLA specific products such as VRLink and the RTI. Encapsulating HLA specific activity within these objects reduces the overall impact to the product as HLA evolves into the future.

The objects interfacing directly with the host include UPDATE HOST GENERATED DATA and PROCESS HLA GENERATED DATA. These objects contain all references to host specific characteristics and functions such as memory mapping, type/unit conversion. Encapsulation of this host specific data within these objects reduces the overall impact to the product as other flight simulators are incorporated into the product.

This logical design could certainly be made more efficient by not following the above stated objectives, but given the volatility of the HLA interface and the adaptability needed to accommodate other host computers it is imperative that this high level compartmentalization be enforced.

At this point COTS software must be considered to reduce cost and risk. The HLA standard has provided an opportunity to vendors to provide products that interface to the HLA network and provide a higher level API to the implementer thus reducing development cost assuming that the product fits into their design requirements. After extensive analysis and prototyping, a product from MÄK Technologies called VRLink was chosen as a COTS software product that would fill the above stated requirements.

VRLink provides an extensive set of classes and methods that eliminate the need to develop software that deals directly with the RTI. In fact the API provided by VRLink closely resembles several objects defined in the logical model. Figure 2 identifies these objects in the logical model. The key components of VRLink that provide the desired functionality are entity publishers and reflectors. The entity reflector tool takes the place of the UPDATE HLA GENERATED DATA object and provides an interface like the HLA GENERATED DATA class as shown on the logical diagram. The reflector maintains a list off all entities and their attributes on the HLA network and provides access to this data through simple methods. The entity publisher tool takes the place of the PROCESS HOST GENERATED DATA object and provides an interface like the HOST GENERATED DATA class as shown on the logical diagram. The publisher maintains and transmits entity information providing a simple interface for updating entity attributes as required. Interactions for firing and detonation of munitions are also included.

Using the VRLink product does not come without drawbacks. First of all there are restrictions on the FOM and its modification. VRLink is based on the RPR FOM therefore when committing to use VRLink you are committing to using the RPR FOM. Also VRLink does not provide all the flexibility of accessing the RTI directly. These restrictions are acceptable for our Tomcat application of the

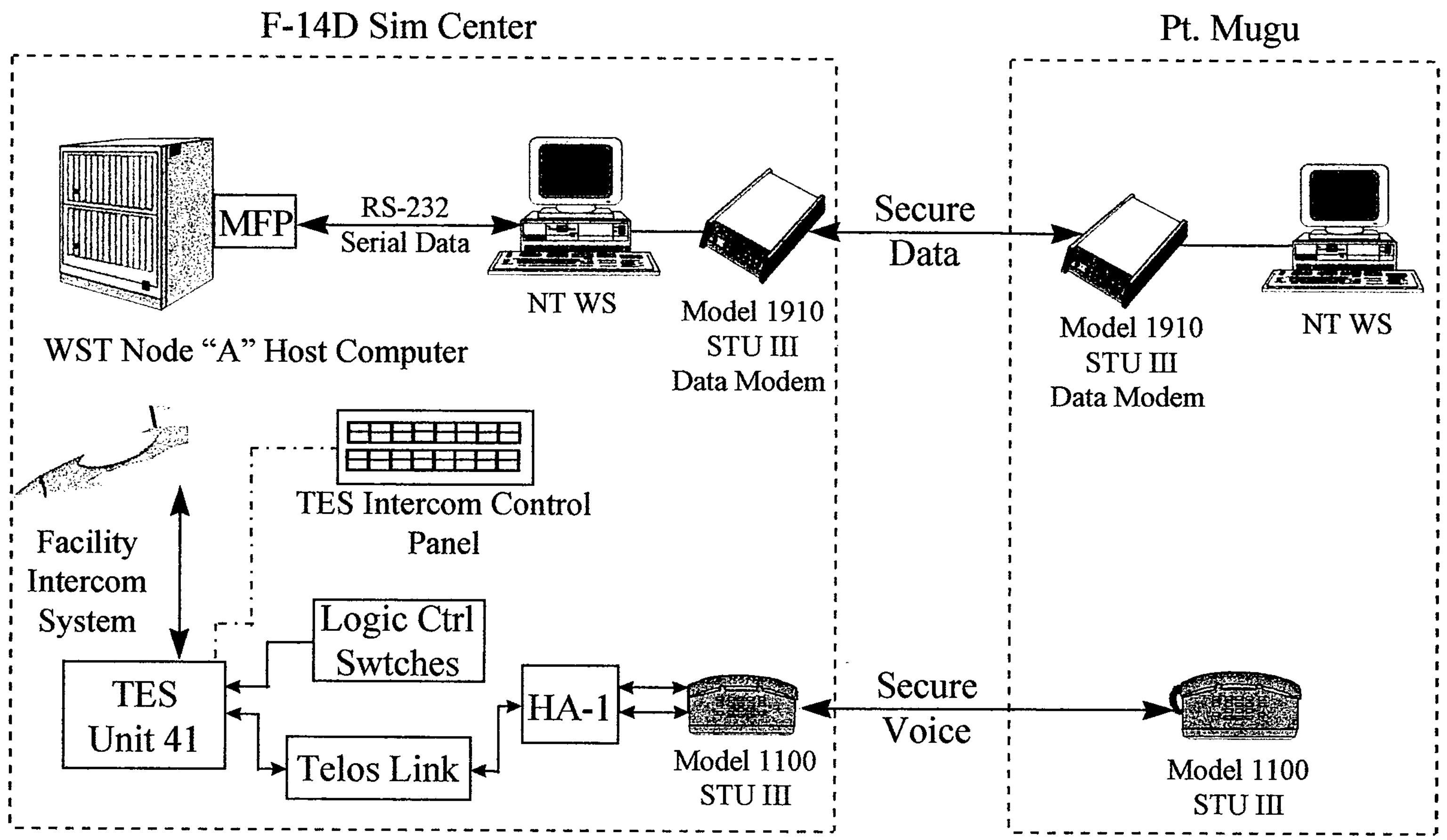
VFTNIU but there are reservations for the long term capability of VRLink to meet our growth objectives. Given the estimated effort saved by using VRLink and a limited budget, we have chosen to use VRLink with the following additional design objective: Use VRLink products in a clearly defined manner such that transition to another HLA access technique has minimal design disruption in the future.

## 2.4. TESTING RESULTS

The VFTNIU was tested using a software program that emulates the inputs that would be transmitted and received from the F-14D trainer. Since a physical connection was never established testing was done with software using shared memory. Shared memory is designed to allow an independent program to transfer information to the main VFTNIU process. Optimally this process would be one that controls I/O equipment that physically send and receive data to the F-14D trainer (such as a HSD device). Dual developed a test driver that loads and unloads shared memory in the same manner as the I/O program would have done. The program is the F-14D Emulation Program. Detailed data on how to run this program are included in Appendix F. Extensive testing and debug of this program were completed in Orlando and demonstrated in Oceana.

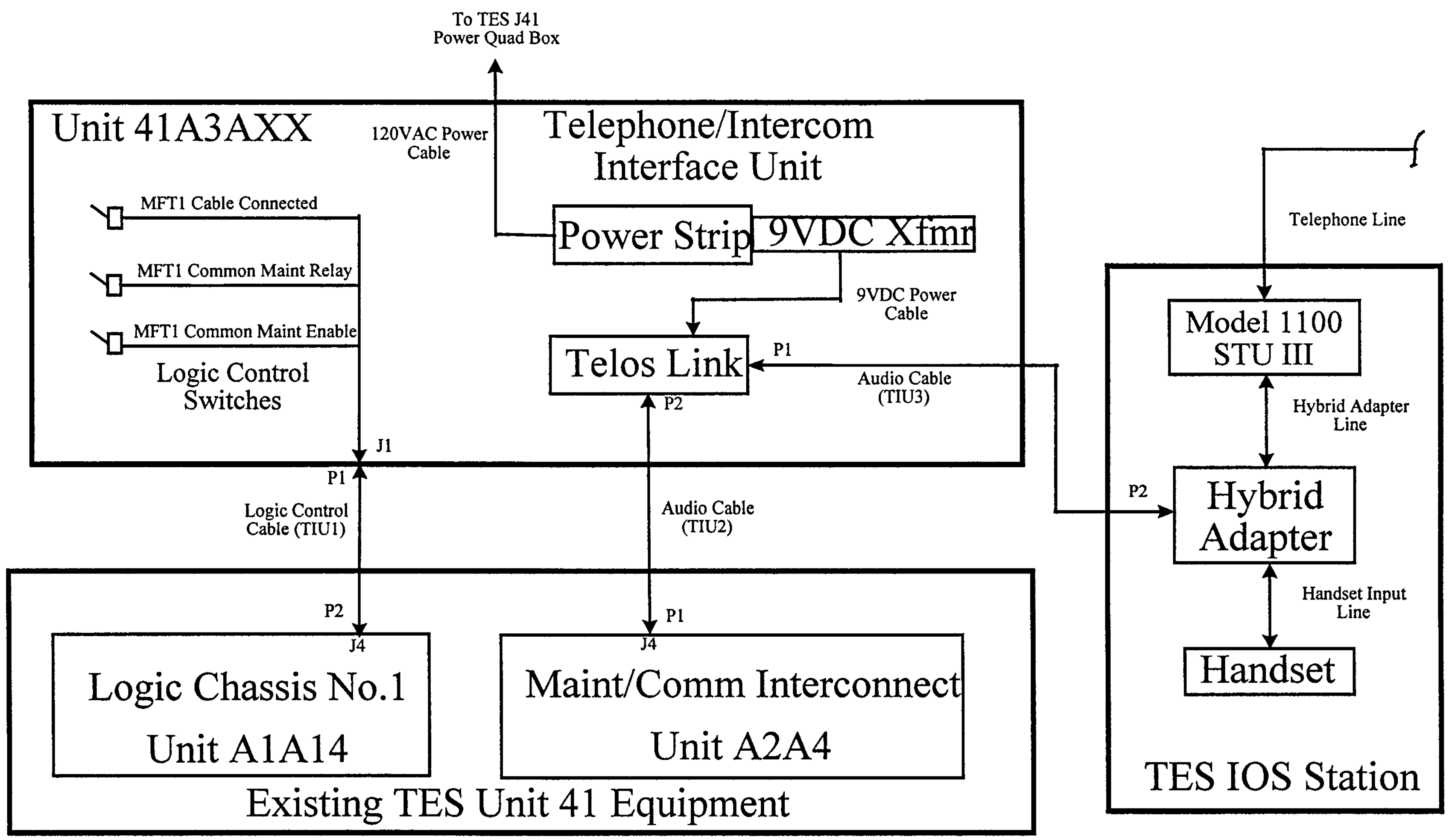
# APPENDIX A

# TOMCAT SOFTWARE UPDATE MODE OVERVIEW


Tomcat Software Update Mode
F-14D Sim Center
Pt. Mugu
MFP
RS-232
Serial Data
NT WS
Model 1910
STU III
Data Modem
WST Node "A" Host Computer
Secure
Data
Model 1910
STU III
Data Modem
NT WS
TES Intercom Control
Panel
Facility
Intercom
System
Logic Ctrl
Swtches
TES
Unit 41
HA-1
Telos Link
Model 1100
STU III
Secure
Voice
Model 1100
STU III

# APPENDIX B

# TOMCAT TELEPHONE/INTERCOM INTERFACE SYSTEM DESIGN



Tomcat Telephone/Intercom Interface System Diagram

# APPENDIX C

# TOMCAT SOFTWARE UPDATE MODE DESIGN

| DWG NO. | SHT | REV |
|---|---|---|
| 9806-0001 | 1 | - |

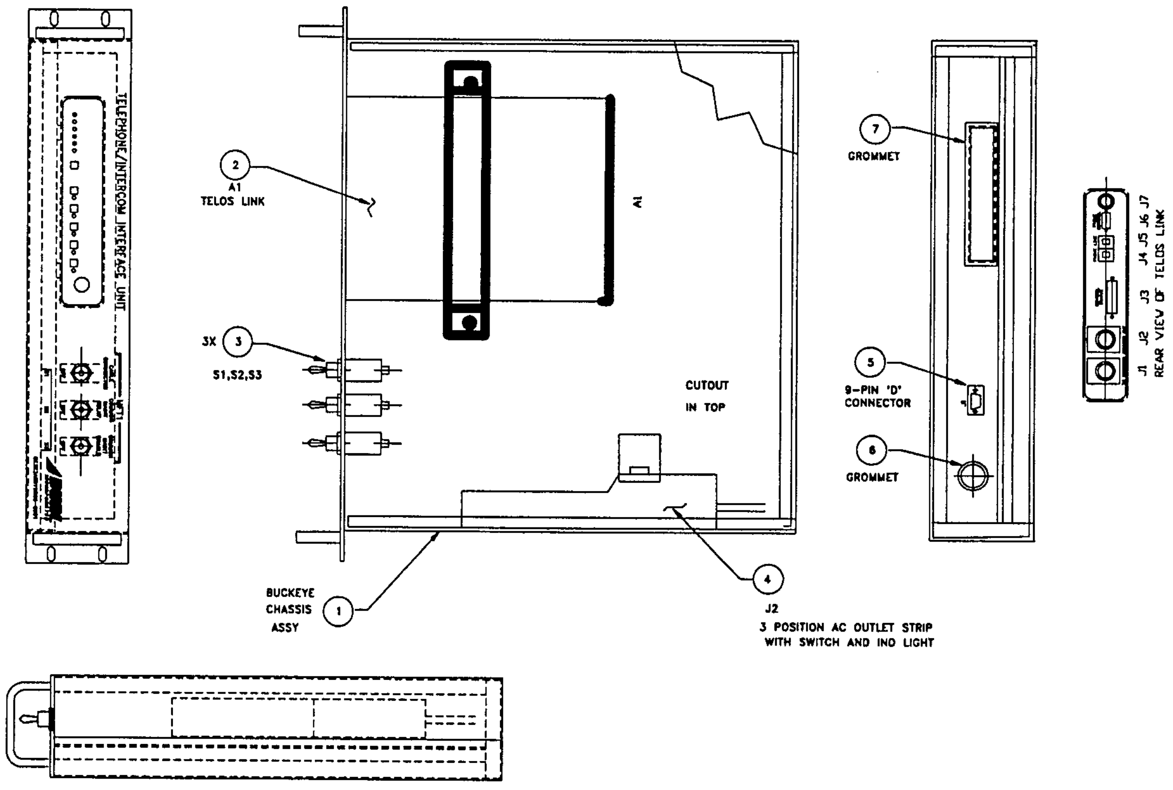
REVISIONS

| REV | DESCRIPTION | DATE | APPROVED |
|---|---|---|---|
| | | | |

NOTES: UNLESS OTHERWISE SPECIFIED:

1. APPLICABLE STANDARDS/SPECIFICATIONS
   A. MIL-STD-100E
   B. MIL-T-31000
3. WORKMANSHIP SHALL BE IAW MIL-STD 454M REQUIREMENT 9.
4. SOLDER IAW MIL-STD-454, REQUIREMENT 5.
5. TAPPED HOLES TO BE FREE OF PRIMER AND PAINT.
6. TERMINATIONS SHALL BE IAW MIL-STD-454M, REQUIREMENT 19.
7. INTERNAL WIRING SHALL BE IAW 454M, REQUIREMENT 69.
8. INSTALL WIRE MARKERS PER MIL-T-23991
9. REFERENCE DESIGNATORS: MARK IAW MIL-STD-130, LOCATE APPROXIMATELY WHERE SHOWN.
10. HARDWARE SUPPLIED WITH Buckeye CHASSIS ASSY.

FRONT PANEL FINISH:
BATTLESHIP GRAY,
NOMENCATURE-WHITE



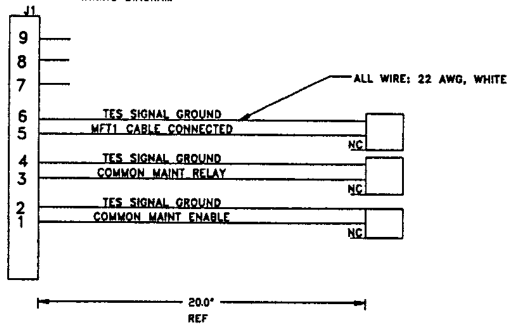
PARTS

J1 - CONNECTOR, 9-PIN 'D'
MALE PIN, SOLDER CUPS
WITH FEMALE SCREWLOCK KIT
(AMP 747904-2 OR EQUIV )

S1,S2,S3 - TOGGLE SWITCH SPST,
OFF-NONE-ON, MINITURE, .25 BUSHING
( C&K 7101SYZQE OR EQUIV )

MP-42131002 Full Chasis 16" DEEP
MP-42128-01 Rear Panel
SM-3516-TBSH Sheet Metal Case W/Top
130-1 Telos link Phone/Intercom
HA-1 Excalibur Hybrid Adapter
20' A/C Electrical Power cord
9806-0001 Switches/Cable Assy
9806-0003 Logic Control Cable
9806-0004 Audio Cable
9806-0005 Audio Cable

WIRING DIAGRAM



| | | |
|---|---|---|
| DO NOT SCALE DRAWING | B.BOTTOM 98/10/02 | DUAL, INC. 36 SKYLINE DRIVE LAKE MARY, FL 32746 |
| THIRD ANGLE PROJECTION | N00421-98-C-1119 | TOMCAT TELEPHONE/INTERCOM INTERFACE UNIT |
| NEXT ASSEMBLY / USED ON — APPLICATION | | D 0E2Z3 9806-0001 - |
| | | SCALE 1/2 000101.DWG SHEET 1 OF 1 |

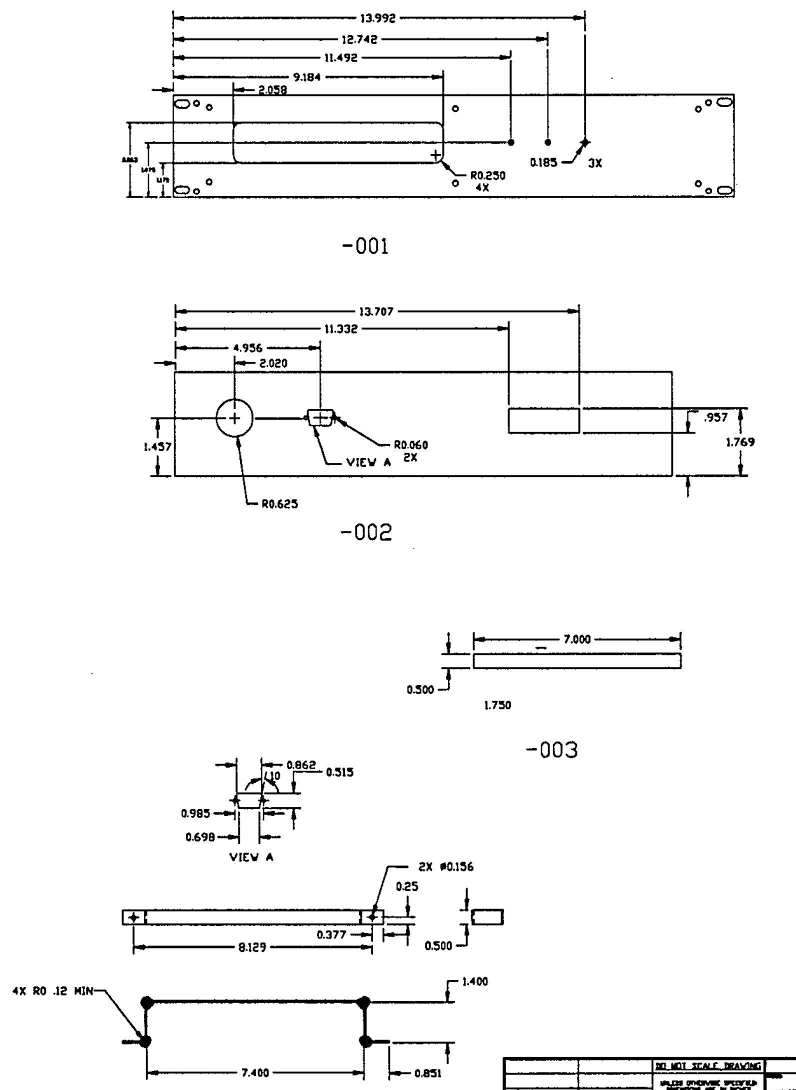
| | | | |
|---|---|---|---|
| 9806-0002 | 1 | - | |

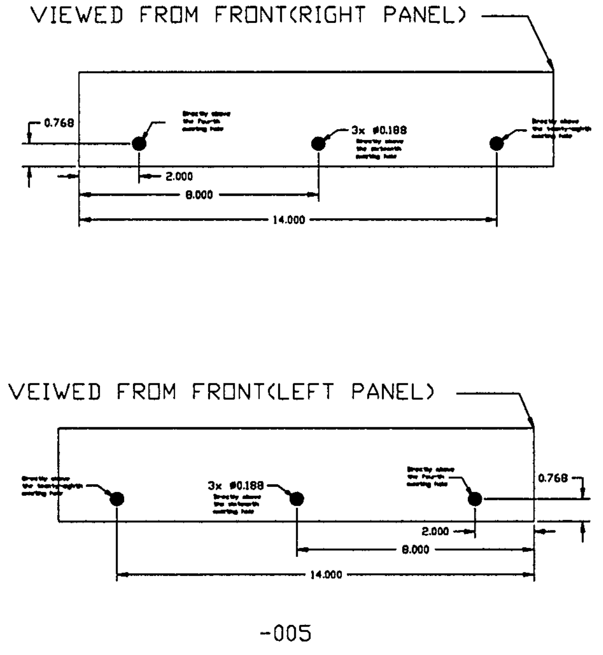
REVISIONS

| REV | DESCRIPTION | DATE | APPROVED |
|---|---|---|---|
| | | | |

NOTES: UNLESS OTHERWISE SPECIFIED:

1. APPLICABLE STANDARDS/SPECIFICATIONS:
   A. MIL-STD-100E.
   B. MIL-T-31000.
2. WORKMANSHIP SHALL BE IAW MIL-STD-454M, REQ. 9.
3. PARTIAL REFERENCE DESIGNATIONS ARE SHOWN. FOR COMPLETE DESIGNATION, PREFIX WITH UNIT NUMBER OR SUBASSEMBLY DESIGNATION OR BOTH.
4. REMOVE BURRS AND BREAK ALL SHARP EDGES.
5. TAPPED HOLES TO BE FREE OF PRIMER AND PAINT.
6. MATERIAL: -001 MAKE FROM ( Buckeye SM Series FRONT PANEL )
   -002 MAKE FROM ( Buckeye SM Series REAR PANEL )
   -003 NEOPRENE RUBBER DUROMETER HARDNESS 25-40, .125 THK STK
   -004 AL ALY, 5052-H32 PER QQ-A-250/8, .090 THK STK.
   -005 MAKE FROM ( Buckeye SM Series SIDE PANELS )

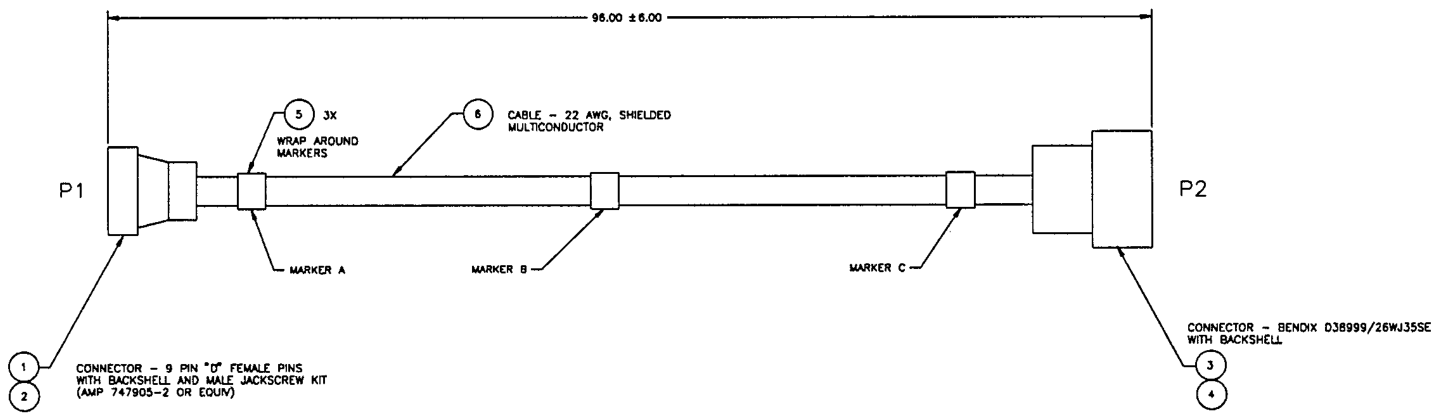




| | | DO NOT SCALE DRAWING | APPROVALS | DATE | | |
|---|---|---|---|---|---|---|
| | | | W. BUSH | 99/9/17 | DUAL INCORPORATED | DUAL, INC. 38 SKYLINE DRIVE LAKE MARY, FL 32746 |
| | | THIRD ANGLE PROJECTION | | | TOMCAT TELEPHONE/INTERCOM INTERFACE DETAIL PANEL | |
| NEXT ASSEMBLY | USED ON | | N00421-98-C-1119 | | D 0E2Z3 | 9806-0002 - |
| APPLICATION | | | | | 1/2 | 000201.DWG 1 OF 1 |

9806-0003 SHT 1 REV -

| REVISIONS | | | |
|---|---|---|---|
| REV | DESCRIPTION | DATE | APPROVED |


96.00 ±6.00
P1
P2
5 3X
WRAP AROUND MARKERS
6 CABLE – 22 AWG, SHIELDED MULTICONDUCTOR
MARKER A
MARKER B
MARKER C
1 2 CONNECTOR – 9 PIN "D" FEMALE PINS WITH BACKSHELL AND MALE JACKSCREW KIT (AMP 747905-2 OR EQUIV)
3 4 CONNECTOR – BENDIX D38999/26WJ35SE WITH BACKSHELL

| MARKER DATA | |
|---|---|
| MARKER A | TIU1 P1 A100-J1 |
| MARKER B | OE2Z3ASSY 9806-0003-01 TIU1 |
| MARKER C | TIU1 P2 41A1A14J4 |

WIRING DIAGRAM

| P1 | | P2 |
|---|---|---|
| 1 | MFT1 COMMON MAINT ENABLE | 44 |
| 2 | TES SIGNAL GROUND | 19 |
| 3 | MFTI COMMON MAINT RELAY | 40 |
| 4 | TES SIGNAL GROUND | 18 |
| 5 | MFT1 CABLE CONNECTED | 51 |
| 6 | TES SIGNAL GROUND | 50 |

DO NOT SCALE DRAWING
THIRD ANGLE PROJECTION
NEXT ASSEMBLY | USED ON
APPLICATION

APPROVALS: B BOTTOM 98/09/23

CONTRACT OR P.A. NO. N00421-98-C-1119

DUAL INCORPORATED
DUAL, INC.
36 SKYLINE DRIVE
LAKE MARY, FL 32746

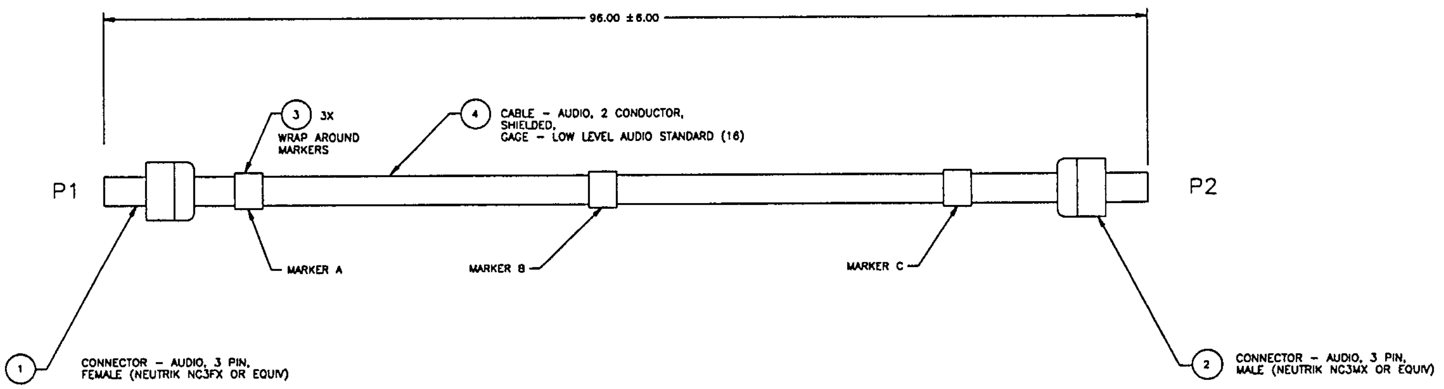
CABLE ASSEMBLY
TIU1
LOGIC CONTROL

SIZE D | CAGE CODE OE2Z3 | DWG NO. 9806-0003 | REV -
SCALE: NONE | CADFILE: 000301.DWG | SHEET 1 OF 1

| DWG NO. | SHT | REV |
|---|---|---|
| 9806-0004 | 1 | - |

REVISIONS

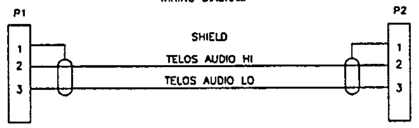
| REV | DESCRIPTION | DATE | APPROVED |
|---|---|---|---|
| | | | |



96.00 ±6.00
P1
P2
3 3X WRAP AROUND MARKERS
4 CABLE - AUDIO, 2 CONDUCTOR, SHIELDED, GAGE - LOW LEVEL AUDIO STANDARD (16)
MARKER A
MARKER B
MARKER C
1 CONNECTOR - AUDIO, 3 PIN, FEMALE (NEUTRIK NC3FX OR EQUIV)
2 CONNECTOR - AUDIO, 3 PIN, MALE (NEUTRIK NC3MX OR EQUIV)

| MARKER DATA | |
|---|---|
| MARKER A | TIU2 P1 A100A1-J1 |
| MARKER B | 0E2Z3ASSY 9806-0004-01 TIU2 |
| MARKER C | TIU2 P2 41A2A4J4 |

WIRING DIAGRAM



| NEXT ASSEMBLY | USED ON |
|---|---|
| APPLICATION | |

DO NOT SCALE DRAWING

THIRD ANGLE PROJECTION

| APPROVALS | DATE |
|---|---|
| B BOTTOM | 98/09/23 |

CONTRACT OR P.O. NO. N00421-98-C-1119

DUAL INCORPORATED

DUAL, INC.
34 SKYLINE DRIVE
LAKE MARY, FL 32746

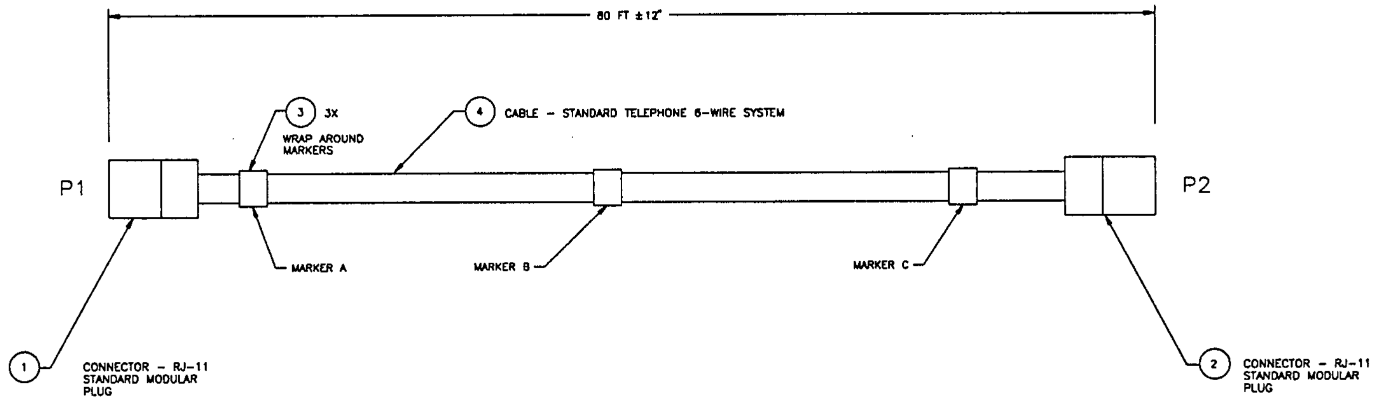
CABLE ASSEMBLY
TIU2
AUDIO

| SIZE | CAGE CODE | DWG NO. | REV |
|---|---|---|---|
| D | 0E2Z3 | 9806-0004 | - |
| SCALE NONE | CADFILE 000401.DWG | SHEET 1 OF 1 | |

9806-0005 SHT 1 REV -

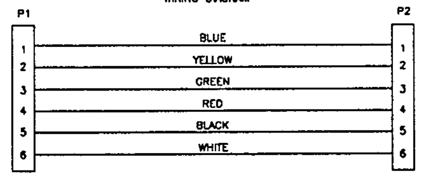
REVISIONS

| REV | DESCRIPTION | DATE | APPROVED |
|---|---|---|---|
| | | | |


80 FT ±12"
P1
P2
3 3X WRAP AROUND MARKERS
4 CABLE - STANDARD TELEPHONE 6-WIRE SYSTEM
MARKER A
MARKER B
MARKER C
1 CONNECTOR - RJ-11 STANDARD MODULAR PLUG
2 CONNECTOR - RJ-11 STANDARD MODULAR PLUG

| MARKER DATA | |
|---|---|
| MARKER A | TIU3 P1 A100A1-J4 |
| MARKER B | 0E2Z3ASSY 9806-0005-01 TIU3 |
| MARKER C | TIU3 P2 PHONE JACK |

WIRING DIAGRAM



DO NOT SCALE DRAWING
THIRD ANGLE PROJECTION
APPROVALS B BOTTOM 98/09/23
CONTRACT NO. N00421-98-C-1119
NEXT ASSEMBLY | USED ON
APPLICATION

DUAL INCORPORATED
DUAL, INC. 38 SKYLINE DRIVE LAKE MARY, FL 32746

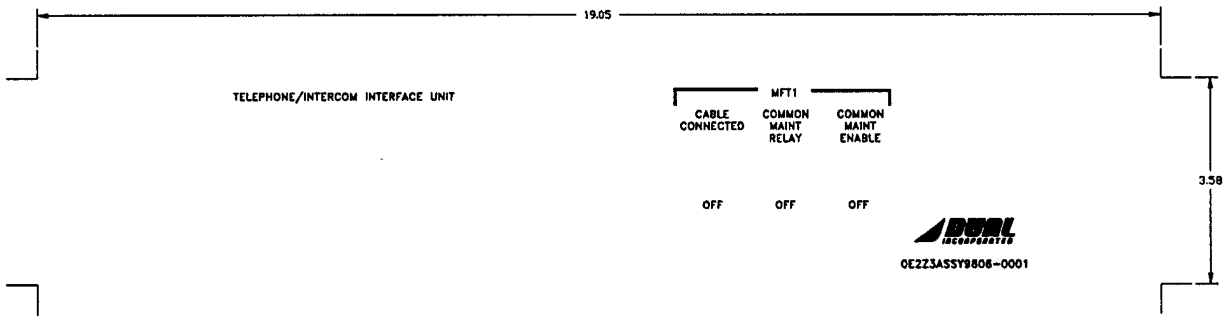
CABLE ASSEMBLY
TIU3
TELEPHONE

SIZE D | CAGE CODE 0E2Z3 | DWG NO. 9806-0005 | REV -
SCALE NONE | 000501.DWG | SHEET 1 OF 1

| DWG NO. | SHT | REV |
|---|---|---|
| 9806-0006 | 1 | - |

REVISIONS

| REV | DESCRIPTION | DATE | APPROVED |
|---|---|---|---|
| | | | |



| | | | | |
|---|---|---|---|---|
| NEXT ASSEMBLY | USED ON | DO NOT SCALE DRAWING | APPROVALS: V. BUSH | DATE: 99/9/27 |
| APPLICATION | | THIRD ANGLE PROJECTION | CONTRACT NO. / P.O. NO.: N00421-98-C-1119 | |

DUAL INCORPORATED — DUAL, INC., 38 SKYLINE DRIVE, LAKE MARY, FL 32746

TOMCAT
TELEPHONE/INTERCOM
INTERFACE UNIT

| SIZE | CAGE CODE | DWG NO. | REV |
|---|---|---|---|
| D | OE2Z3 | 9806-0006 | - |
| SCALE 1/2 | | 000601.DWG | SHEET 1 OF 1 |

# APPENDIX D

# TOMCAT SOFTWARE UPDATE MODE USERS GUIDE

# Users Notes for Pt. Mugu

These instructions are for setting up the Dell at Pt. Mugu as a remote computer connection to the Encore computer. In this configuration Pt. Mugu establishes a secure data link with the computer at the F14D TES OPCOM via analog phone lines and STU III modems. pcAnywhere is used to link this computer to the Intergraph computer at the F14D trainer such that the Pt. Mugu computer can operate the Intergraph remotely. The Intergraph at the F14D is connected to the Encore via com2. This effectively allows Pt. Mugu to login to the Encore and perform file transfers and etc..

1. Open pcAnywhere (There is a shortcut labeled pcAnywhere that should be visible when the computer starts, otherwise find it in the start menu (start >> programs >> pcAnywhere))

2. Select "Remote Control"

3. Select the type of connection desired by double clicking on the icon SECTEL1500. This identifies com1 as the data port to which the secure data communications device is to be connected.

4. At this point the pcAnywhere software will wait for a carrier detect signal indicating that the modems have established a connection.

5. Now establish the modem connection. Dual used two SECTEL model 1500's. This involved establishing a clear voice connection then pressing data and secure at both SECTELs. Upon completing the secure link the pcAnywhere software will "wake up".

5. The Pt. Mugu computer will provide an emulation of the PC at the F14D trainer.

6. At this point the icon on the desktop labeled "Login to F14 Trainer" should be double clicked. This starts a terminal emulation program provided as part of Windows. From here the emulation window acts like the OPCOM terminal 33A3.

Note : pcAnywhere manual is provided on the desktop.

# TES OPCOM User Notes

These instructions are for setting up the Intergraph PC at the TES IOS as a "gateway" from a remote computer (Pt. Mugu) to the Encore computer. In this configuration Pt. Mugu establishes a secure data link with this computer via analog phone lines and secure data modems. Then pcAnywhere is used to link this computer to Pt. Mugu such that the Pt. Mugu computer can operate this computer remotely. This computer is connected to the Encore TES opcom port through COM2. This effectively allows Pt. Mugu to login to the Encore and perform file transfers and etc..

1. Launch pcAnywhere from the desktop.

2. Select "Be a Host PC" (The Pt. Mugu computer is the remote control, and the Intergraph [this one] is the host).

3. Select the type of connection desired by double clicking on the icon SECTEL1500. This identifies com1 as the data port to which the secure data communications device is to be connected.

4. At this point the pcAnywhere software will wait for a carrier detect (modem connected to another modem).

5. Pt. Mugu must start their pcAnywhere software as a remote control.

6. At this point a secure data connection must be made with the modems.

7. Upon carrier detect, the Pt. Mugu pcAnywhere software will connect with this computer and allow for remote control.

8. At this point the icon on the desktop labeled "Login to F14 Trainer" should be double clicked. This starts a terminal emulation program provided as part of Windows. From here the emulation window acts like the OPCOM terminal 33A3.

Note : pcAnywhere manual is provided on the desktop.

# APPENDIX E

## VERSATILE FLIGHT TRAINER NETWORK INTERFACE UNIT DESIGN

# VFTNIU SOFTWARE DETAILED DESIGN

The detailed software design will be described below by providing an overview of the VFTNIU runtime environment environment and its current status. Then detailed design information about the program will be provided followed by listings of the code. Very detailed design information will be provided as comments in the code.

Figure 1 VFTNIU SOFTWARE DESIGN HIERARCHICAL STRUCTURE DIAGRAM shows the VFTNIU main program and its subordinate classes and procedures.

**VFTNIU.cxx** - contains the main program for the VFTNIU. This main program controls the iteration rate of the program as well as the keyboard input. The keypad is polled here so as not to interfere with iteration timing. The main program invokes two main procedures called ProcessData and UpdateData which respectively control the receiving and transmition of data on the HLA network.

**ProcessHLAData.h, ProcessHLAData.cxx** - contains the procedure ProcessData. This procedure manages the Input of data from the HLA network. This is done by maintaining a linked list of entities that are currently being processed and comparing this against the new list that is received every iteration. All interfaces to the HLA network are done through the HLAInterface class. This isolates the management funtion of this procedure from the implementation details of interfacing to the HLA network.

**UpdateHostData.h, UpdateHostData.cxx** - contains the procedure UpdateData. This procedure manages the output of data to the HLA network. This is done by maintaining a list of possible entities that can be transmitted. First the procedure getNewEntityStatus from the hostDataExtraction class is invoked to determine the current status of entities coming from the host trainer. New entities initiate the creation of an entity publisher while entities removed from active status result in the deletion of the associated publisher. All interfaces to the HLA network are done through the HLAInterface class. This isolates the management funtion of this procedure from the implementation details of interfacing to the HLA network.

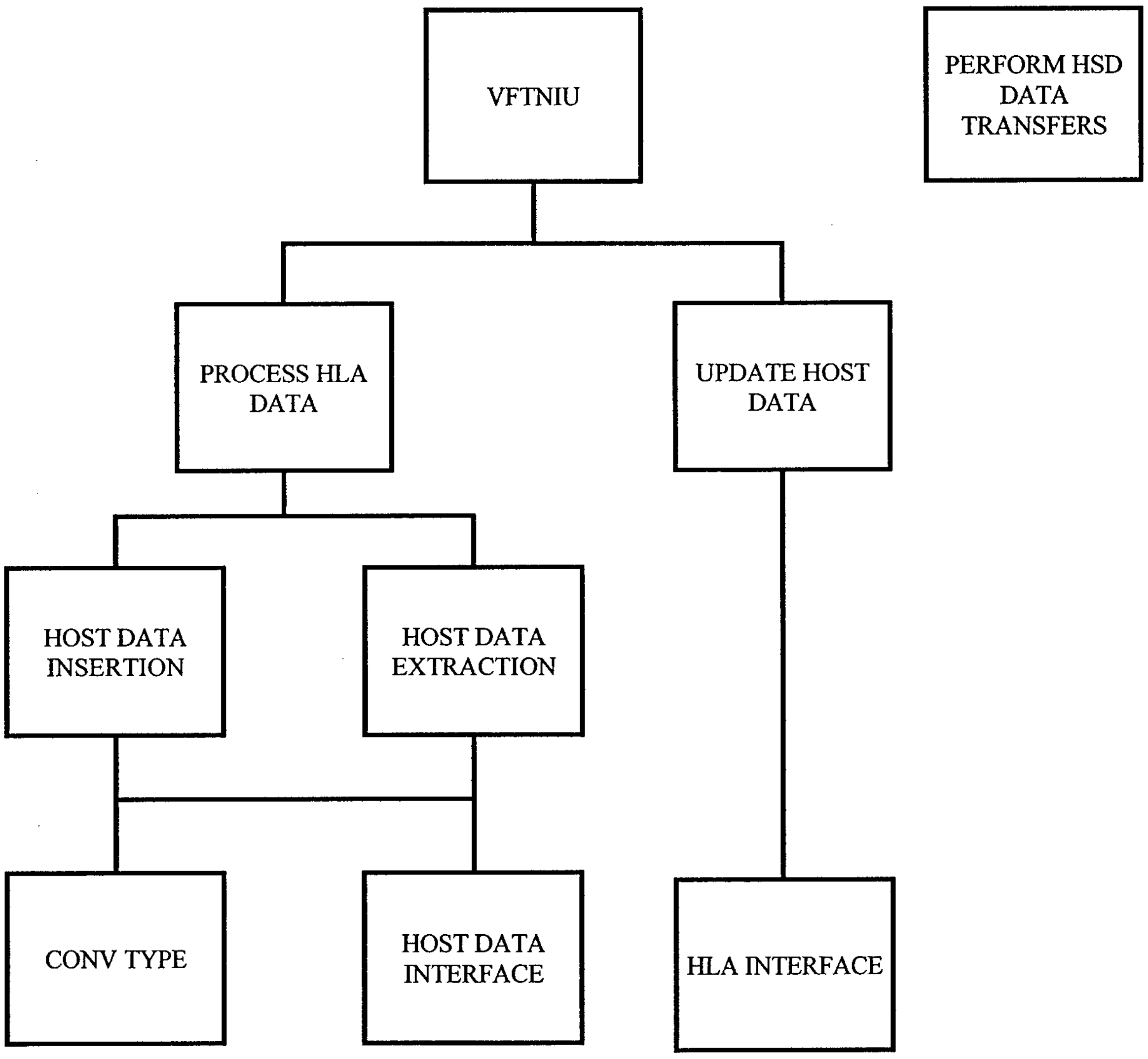
**HLA_Interface.h, HLA_Interface.cpp** - is a class of procedures that directly interface to the HLA network. Member functions pass back and receive entity information independant of the technique in which the HLA network is accessed. This is where the VRLink product is used. The VRLink API is encapsulated inside this class to increase mainainability.

**HostDataExtraction.h, HostDataExtraction.cpp** - is a class that allows for access of host data from sharred memory using an internal entity number. This internal number for each entity maps it to a specific entiy from the host simulator. The physical memory location for each entity are maintained in the class hostDataInterface.

**HostDataInsertion.h, HostDataInsertion.cpp** - is a class that allows for transmission of host data to sharred memory. The entity type is used to map entity data to a specific entity on the host simulator. Entity types that do not have a match are ignorred. The physical memory location for each entity are maintained in the class hostDataInterface.

**hostDataInterface.h, hostDataInterface.cpp** - is a class that allows for host data to be

# VFTNIU SOFTWARE DESIGN HIERARCHICAL STRUCTURE DIAGRAM



transferred based on its displacement from the beginning of sharred memory areas. The displacements are maintained in terms of entity name and type of data. The constuctor for this class defines the physical address for each entity and type. This will be the only code required to change when the physiscal location of the data is changed in sharred memory.

**convTypes.h, convTypes.c** - is a set of procedures that perfom coversion to/from Gould format from/to IEEE (Microsoft) format.

```
//
// Vftniu.cxx
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//
/*
This program provides access to an HLA network for a standalone (host) simulator.
Interface to the host simulator is physically implemented through interface hardware.
This program retrieves and loads memory consistent with the requirements of this interface.

Classes and methods particular to the interface are developed as required.

The first simulator interfaced is the F14 Tomcat simulator via a Gould HSD device.
Unique class structures and methods have been developed to support this implementation.
*/

#include "stdio.h"
#include "ProcControl.h"
#include "HLAInterface.h"
#include "UpdateHostData.h"
#include "ProcessHLAData.h"
#include "vftniuTypes.h"


int keybrdTick(void);


main()

{

   // Initialize VR-Link time.
   DtTimeInit();
   DtTime dt = 0.05;     // 20 hz
    DtTime simTime = 0;


    // define an instance of HLAInterface
    HLAInterface HLAGate;

    // set up pointer to object of class
    HLAInterface* pHLAInt;
    pHLAInt = &HLAGate;

    UpdateHostData myUpdateHostData;
    ProcessHLAData  myProcessHLAData;

   // Enter main simulation loop
   int forever = 1;
   while (forever)      // loop until quit key is pressed
   {

        // Monitor keyboard for IOS instructions
        if (keybrdTick() == -1)
            break;

        // update simulation time for VRLink
        DtSetSimTime(simTime);
        // get data off the HLA net and process it
        myProcessHLAData.ProcessData(pHLAInt);

        // Extract data from host memory and put it on the HLA net
        myUpdateHostData.UpdateData(pHLAInt);

        // add iteration duration to simtime
        simTime     += dt;

        // sleep till dt time elapsed since last sleep
        //printf("sleep ");
        DtSleep(simTime - DtGetElapsedRealTime());

    }

    // Terminate Program
    return 0;

}


int keybrdTick()
{
    char *keyPtr = DtPollInputLine();
    if (keyPtr && (*keyPtr == 'q' || *keyPtr == 'Q'))
        return -1;
    else
        return 0;
}
```

```
//
// VftniuTypes.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//


#ifndef vftniuTypes_H_
#define vftniuTypes_H_

#include "exerciseConn.h"
#include "reflEntList.h"
#include "ProcControl.h"
#include "entitySR.h"
#include "reflectedEnt.h"
#include "fireInter.h"
#include "topoView.h"
#include "EntityTypes.h"
#include "entityPub.h"


// AIRCRAFT

// Define the type for the host
static DtEntityType f14(DtPlatform, DtPlatformDomainAir,
DtUnitedStates, DtFighter, DtF18, 0, 0);      // f14 not available

// Define the type for the simulated dome aircraft to be mapped into ownship
static DtEntityType f14Dome(DtPlatform, DtPlatformDomainAir,
DtUnitedStates, DtFighter, DtF18, 0, 7);      // f14 not available


static DtEntityType su25(DtPlatform, DtPlatformDomainAir,
DtUnionOfSovietSocialistRepublics, DtAttack, DtSU25, 0, 0);

static DtEntityType c130(DtPlatform, DtPlatformDomainAir,
DtUnitedStates, DtCargo, DtC130, 0, 0);

static DtEntityType a10(DtPlatform, DtPlatformDomainAir,
DtUnitedStates, DtAttack, DtA10, 0, 0);

static DtEntityType ah64(DtPlatform, DtPlatformDomainAir,
DtUnitedStates, DtAttackHelicopter, DtAH64, 0, 0);

static DtEntityType ussrAttachHelo(DtPlatform, DtPlatformDomainAir,
DtUnionOfSovietSocialistRepublics, DtAttackHelicopter, DtMI28, 0, 0);

static DtEntityType ussrBomber(DtPlatform, DtPlatformDomainAir,
DtUnionOfSovietSocialistRepublics, DtBomber, DtTU26, 0, 0);


// MISSLES

// all green large pointy missiles (anti- air)
static DtEntityType MissleLargeGreen(DtMunition, DtAntiAir,
DtUnionOfSovietSocialistRepublics, DtMunitionGuided, 0, 0, 0);


// green small big rudders (anti armor and anti guided)
static DtEntityType MissleSparrow(DtMunition, DtAntiArmor,
DtUnionOfSovietSocialistRepublics, DtMunitionGuided, 0, 0, 0);


// big white missles  (all us anti air  Unguided AND GUIDED)
static DtEntityType MissleLargeWhite(DtMunition, DtAntiAir,
DtUnitedStates, DtMunitionUnguided, DtSidewinder, 0, 0);

// big white missles specific for domeType
static DtEntityType domeMissileType(DtMunition, DtAntiAir,
DtUnitedStates, DtMunitionUnguided, DtSidewinder, 0, 7);




#define max_host_entities 5

struct entStat
{
    int status[max_host_entities];          // 0 = inactive, 1= active
    DtEntityType type[max_host_entities];
};

struct HLAentList
{
    int status;          // 0 = inactive, 1= active
    char *id;
    struct HLAentList *next;
};


struct entData
{
    char entId[20];
    DtEntityType type;
    DtVector position;
    DtVector velocity;
    DtVector acceleration;
    DtTaitBryan orientation;
    DtVector rotationalVelocity;
    char markText[20];
};
enum dataType {Int, Double, Float, Bool};

struct datamap {
    int disp;
    dataType type;
};

#define maxNumF14Missiles 10
#define numberOfEntityParams 16

enum entityParams    {
                        status, posX, posY, posZ,
                        velX, velY, velZ,
                        accelX,accelY,accelZ,
                        orientX,orientY,orientZ,
                        rotVelX,rotVelY,rotVelZ
                    };

#endif
```

```
//
// ProcessHLAData.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

#include "HLAInterface.h"

class ProcessHLAData
{
public:
    ProcessHLAData();
    void ProcessData(HLAInterface* pHLAInt);
};
```

```
//
// ProcessHLAData.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
Maintain entities that exist on the HLA network.  This is done
by accessing entity data using the HLA Interface package.  This
data is then processed and if necessary loaded into F14 shared data area
using the HostDataExtraction package.
*/


#include "stdio.h"
#include "ProcessHLAData.h"
#include "vftniuTypes.h"

#include "entityPub.h"      // defines DTVector, DtTaitBryan
//#include "hostDataExtraction.h"
#include "hostDataInsertion.h"

// define a linked list for maintaining HLA entity status from last iteration.
//  This is necessary so that addition and deletion of entities can be detected

HLAentList start, *myHLAEntity, *previous;

HostDataInsertion myHostDataInsertion;

ProcessHLAData::ProcessHLAData()
{
    // set up a linked list
    start.next = NULL;          // empty list
    myHLAEntity = &start;       // point to first entity

}

void ProcessHLAData::ProcessData(HLAInterface* pHLAInt)
{


    // Declare entity structure array
    entData entityData;


    DtReflectedEntity*  myEntity;


    // clear status flag in each entity (locally maintained)
    myHLAEntity = &start;   // set pointer to first entity
    while (myHLAEntity)
    {
        myHLAEntity->status = 0;
        myHLAEntity = myHLAEntity->next;    // move to next entity
    }


    // refresh HLA entity list and get address of first entity
    myEntity = pHLAInt->refreshHLAEntities();
```

```
// for each entity on the HLA network
while (myEntity)    // while pointer passed back is not null
{
    //  get entity data from hla net
    entityData = pHLAInt->getEntityData(myEntity);


    int entityFound;    // flag for identifying new entities
    entityFound = 0;


    // search for match in local entity list
    myHLAEntity = &start;   // set pointer to first entity
    int done;
    done = 0;
    while (!done)
    {
        // if entity found
        if (myHLAEntity->id == entityData.entId)
        {
            // update entity data
            entityFound = 1;
            myHLAEntity->status = 1;    // set local entity status to active
            done = 1;
        }
        if (myHLAEntity->next == NULL)
            done = 1;
        else
            myHLAEntity = myHLAEntity->next;    // move to next entity
    }


    // if new entity
    if (!entityFound)
    {
        // append entity to linked list
        myHLAEntity->next = (struct HLAentList*) malloc(sizeof(struct HLAentList));

        // move pointer to new entity
        myHLAEntity = myHLAEntity->next;

        myHLAEntity->status = 1;    // set local entity status to active
        myHLAEntity->id = entityData.entId; //
        myHLAEntity->next = NULL;

    }

    // pass entity data on and let host insertion map it to host (if applicable)
    myHostDataInsertion.putEntityData(entityData);


    // get address of next entity in HLA network
    myEntity = pHLAInt->getNextHLAEntity(myEntity);

}   // for all entities in HLA network



// if local entities are no longer on HLA net then delete them
myHLAEntity = start.next;
```

```
    previous = &start;
    while (myHLAEntity)
    {
        // if entity is no longer active
        if (myHLAEntity->status == 0)
        {
            // update pointers, delete and increment to next entity
            previous->next = myHLAEntity->next;
            free (myHLAEntity);
            myHLAEntity = previous->next;
        }
        else
        {
            // increment to next entity
            myHLAEntity = myHLAEntity->next;
            previous = previous->next;
        }
    }


    // print local list for degug
    /*
    myHLAEntity = &start;
    while (myHLAEntity)
    {
        printf("/n local entity id = %s  status = %d",myHLAEntity->id, myHLAEntity->status)
;
        myHLAEntity = myHLAEntity->next;
    }
    */
      // Get interactions

      // Convert and load HLA interaction information into native host format

      // Signal interface process that data is ready for shipment.

};
```

```
//
// UpdateHostData.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

#include "HLAInterface.h"

class UpdateHostData
{
public:


    void UpdateData(HLAInterface* pHLAInt);

};
```

```
//
// UpdateHostData.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
This package manages the movement of entity and interaction information
from the Host to HLA.  Specifically this routine maintains entity status and
requests creation and deletion of entitties on the HLA network.  All entity
information updates are controlled from this routine.
*/

#include "UpdateHostData.h"

#include "vftniuTypes.h"
#include "stdio.h"


// maintain pointers fro all entities
DtEntityPublisher* pentityPublisher[max_host_entities];

// maintain status of entities
entStat oldEntityStatus, newEntityStatus;

HostDataExtraction myHostDataExtraction;

void UpdateHostData::UpdateData(HLAInterface* pHLAInt)
{
    int i;

    // Declare an entity structure array
    entData entityData;

    // command Host Extraction routine to prepare for an iteration by copying
    // host data from the DMA memory area to a buffer
    myHostDataExtraction.refreshData();

    // update entity status array
    newEntityStatus = myHostDataExtraction.getNewEntityStatus();

    // now test for differences in entity status and add/delete entity publishers as necess
ary
    for  (i=0;i<max_host_entities;i++)
    {
        if (newEntityStatus.status[i] == 1 & oldEntityStatus.status[i] == 0)
            // create publisher and keep pointer
        {
            printf("\n ** new entity publisher created \n");
            pentityPublisher[i] = pHLAInt->CreatePublisher(newEntityStatus.type[i]);
        }
        if (newEntityStatus.status[i] == 0 & oldEntityStatus.status[i] == 1)
        {
            printf("\n ** entity publisher deleted \n");
            pHLAInt->DeletePublisher(pentityPublisher[i]);
        }
    }


    // for each active entity extract and load host data
```

```
        for  (i=0;i<5;i++)
    {
        if (newEntityStatus.status[i] == 1)
        {
            // get/convert entity data from host
            entityData = myHostDataExtraction.getEntityData(i);

            // ship data out on HLA net
            pHLAInt->Publisher(pentityPublisher[i],entityData);
        }
    }

    // save for next iteration
    oldEntityStatus = newEntityStatus;



      // Interpret, Convert and load host generated interaction data into HLA compatible st
ructure

      // Load host entity parameters into entity publisher

      // Generate appropriate host interactions

}
```

```
//
// HLA_Interface.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
Interfaces to HLA network via VrLink
*/

#ifndef HLAInterface_H_
#define HLAInterface_H_

#include "vftniuTypes.h"
#include "hostDataExtraction.h"

class HLAInterface
{
    public:

    DtEntityPublisher* CreatePublisher(DtEntityType entityType);

    void Publisher(DtEntityPublisher* pentityPublisher,entData entityData);

    void DeletePublisher(DtEntityPublisher* pentiyPublisher);



    // update reflected entity list and pass back first entity address
    DtReflectedEntity* refreshHLAEntities();

    // get address of next entity  (null retured if no more)
    DtReflectedEntity* getNextHLAEntity(DtReflectedEntity*);

    //  get entity data from hla net
    entData  getEntityData(DtReflectedEntity*);

};

#endif
```

```
//
// HLA_Interface.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
Interfaces to HLA network via VrLink
*/
#include "hostStructs.h"

#include "HLAInterface.h"   // class header
#include "stdio.h"
//#include "vftniuTypes.h"

//  Create HLA connection
static char *execName        = "VR-Link";
char *fedName         = "VFTNIU";
static DtExerciseConn exConn(execName, fedName);

//Create reflected entity list
// (will maintain list of all entities on HLA network and current parameters)
DtReflectedEntityList rel(&exConn);


DtEntityPublisher* HLAInterface::CreatePublisher(DtEntityType entityType)
{

    // Create entity publisher (will maintain host entity parameters)
    DtEntityPublisher* pentityPub = new  DtEntityPublisher(entityType, &exConn,
//                  DtDrDrmRvw, DtForceFriendly,
                    DtDrOther, DtForceFriendly,
                    DtEntityPublisher::guiseSameAsType());

    return pentityPub;

}


void HLAInterface::DeletePublisher(DtEntityPublisher* pentiyPublisher)
{

    delete pentiyPublisher;

    return;
}


void HLAInterface::Publisher(DtEntityPublisher* pentityPublisher,
                             entData entityData)
{

    // pointers to state repository, where we can set data
    DtEntityStateRepository *esrpub;

    // define pointer to state repository
    esrpub = pentityPublisher->entityStateRep();
//  printf(" Update HLA entity #%s ",esrpub->entityId().string());

    // in geocentric
    esrpub->setLocation(entityData.position);
    esrpub->setVelocity(entityData.velocity);
    esrpub->setAcceleration(entityData.acceleration);
    esrpub->setOrientation(entityData.orientation);
    esrpub->setRotationalVelocity(entityData.rotationalVelocity);
    esrpub->setMarkingText(entityData.markText);

    // Call tick, which insures that any data that needs to be
    // updated is sent.
    pentityPublisher->tick();

};



DtReflectedEntity* HLAInterface::refreshHLAEntities()

{
    // Drain input from HLA network
    exConn.drainInput();

    // pass back first entity address
    return rel.first();

}


DtReflectedEntity* HLAInterface::getNextHLAEntity(DtReflectedEntity* pnextEntity)

{
    // update pointer
    pnextEntity = pnextEntity->next();

    return pnextEntity;
}


entData HLAInterface::getEntityData(DtReflectedEntity* pnextEntity)

{
    entData entityData;     // for loading and return

    // Grab its state repository, where we can inspect its data
    DtEntityStateRepository *esr = pnextEntity->entityStateRep();

    // load entity id
    strcpy (entityData.entId, pnextEntity->id().string());

    // load entity position from sharred memory
    entityData.position[DtX] = esr->location()[0];
    entityData.position[DtY] = esr->location()[1];
    entityData.position[DtZ] = esr->location()[2];
    //printf("Pos of entity geocentric x,y,z: %.3f %.3f %.3f \n",
    //esr->location()[0],esr->location()[1], esr->location()[2] );

    entityData.velocity[DtX] = esr->velocity()[0];
    entityData.velocity[DtY] = esr->velocity()[1];
    entityData.velocity[DtZ] = esr->velocity()[2];
    //printf("Velocity of entity x,y,z: %.3f %.3f %.3f \n",
    //esr->velocity()[0],esr->velocity()[1], esr->velocity()[2] );
    // load acceleration
    entityData.acceleration[DtX] = esr->acceleration()[0];
    entityData.acceleration[DtY] = esr->acceleration()[1];
    entityData.acceleration[DtZ] = esr->acceleration()[2];

    // load orientation with geocentric reference
    entityData.orientation.setPsi(esr->orientation().psi());
    entityData.orientation.setTheta(esr->orientation().theta());
    entityData.orientation.setPhi(esr->orientation().phi());


    entityData.rotationalVelocity[DtX] = esr->rotationalVelocity()[0];
    entityData.rotationalVelocity[DtY] = esr->rotationalVelocity()[1];
    entityData.rotationalVelocity[DtZ] = esr->rotationalVelocity()[2];

    entityData.type = esr->entityType();

    return entityData;
}




      // Get interactions
      // Convert and load HLA interaction information into native host format

      // Signal interface process that data is ready for shipment.
```

```
//
// HostDataExtraction.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

#ifndef HostDataExtraction_H_
#define HostDataExtraction_H_

#include "vftniuTypes.h"

class HostDataExtraction
{
public:

    HostDataExtraction();
    ~HostDataExtraction();

    void refreshData();

    entStat getNewEntityStatus();

    entData getEntityData(int entno);

private:


};

#endif
```

```
//
// HostDataExtraction.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
This routine uses low level data mapping found in hostDataInterface
to load meaningful groupings of data such as entity data.
*/


#include "stdio.h"

#include "hostDataExtraction.h"
#include "geodCoord.h"
#include "Euler.h"
#include "LibMatrix.h"
#include "topoCoord.h"

#include "hostDataInterface.h"
#include "convType.h"

#include <iostream.h>

hostDataInterface myHostDataInterface;

HostDataExtraction::HostDataExtraction()
{

    // create and map to sharred memory
    myHostDataInterface.createSharredMemory();
    myHostDataInterface.mapSharredMemory();


}

HostDataExtraction::~HostDataExtraction()
{

    myHostDataInterface.closeSharredMemory();

}



void HostDataExtraction::refreshData()
{

    // refresh local input buffer from sharred memory
    myHostDataInterface.refreshF14Data();

};

// look into host memory and define entity status and type
// this will provide the mapping of specific memory locations
// to certain entity types
entStat HostDataExtraction::getNewEntityStatus()
{
```

```
// NOTE:
    // elememt 0 = ownship
    // element 1-4 = ownship missiles

    entStat newEntStat;

    //  determining status of entities
    // check F14 status (status[0])
    if (myHostDataInterface.getInt(
    myHostDataInterface.F14DataMap[status].disp) == 1)
    {
        newEntStat.status[0] = 1;
    }
    else
    {
        newEntStat.status[0] = 0;
    }


    // missiles will be active based on movement
    static double lastMissilePos = Gould_to_IEEE_dbl(
        myHostDataInterface.getDouble(
        myHostDataInterface.F14MissileDataMap[posX][1].disp));

    if (lastMissilePos == Gould_to_IEEE_dbl(
    myHostDataInterface.getDouble(
    myHostDataInterface.F14MissileDataMap[posX][1].disp)))
        newEntStat.status[1] = 0;
    else
        newEntStat.status[1] = 1;

    // update last pos for next iteration
    lastMissilePos = Gould_to_IEEE_dbl(
        myHostDataInterface.getDouble(
        myHostDataInterface.F14MissileDataMap[posX][1].disp));



    // other missiles are currently inactive
    newEntStat.status[2] = 0;
    newEntStat.status[3] = 0;
    newEntStat.status[4]=  0;

    // initialize type
    // for now 0 = ownship and the remaining will be ownsip missiles
    newEntStat.type[0] = f14;
    newEntStat.type[1] = MissleLargeWhite;
    newEntStat.type[2] = MissleLargeWhite;
    newEntStat.type[3] = MissleLargeWhite;
    newEntStat.type[4] = MissleLargeWhite;



    return newEntStat;
};


// look into host memory and define entity status
entData HostDataExtraction::getEntityData(int entNo)
{
    int i;

    // define structure to be passed back
    entData entityData;

    // generic data map
    struct datamap DataMap[numberOfEntityParams];


    // load specific data map in based on entity number
    if (entNo == 0)
    {
        for(i=0; i<numberOfEntityParams; i++)
        DataMap[i] = myHostDataInterface.F14DataMap[i];

    strcpy(entityData.markText, "F14 Tomcat");

    }

    if (entNo == 1)
    {
        for(i=0; i<numberOfEntityParams; i++)
        DataMap[i] = myHostDataInterface.F14MissileDataMap[i][1];

    strcpy(entityData.markText, "F14 Mis1");

    }


    // add other entNo's as required ...


/*
    // temp test data using topo reference system
    double spacing = .0002;
    static double lat[max_host_entities] = {35.699760,35.699760,35.699760,35.699760,35.6997
60};
    static double lng[max_host_entities] = {-121.326577,
                            -121.326577 + spacing,
                            -121.326577 + 2*spacing,
                            -121.326577 + 3*spacing,
                            -121.326577 + 4*spacing};
    static double alt[max_host_entities] = {1230,1230,1230,1230,1230} ;
    double myHeading = 0, myPitch = 0, myRoll = 0;

    // move entity north (test only)
    lat[entNo] = lat[entNo] + .0001;

    // create geodetic position from
    DtGeodeticCoord mygeod(DtDeg2Rad(lat[entNo]),
                    DtDeg2Rad(lng[entNo]),
                    alt[entNo]);
    // covert geodetic to geocentric
    DtVector mygeoc = mygeod.geocentric();
*/
    // load entity position from sharred memory
    entityData.position[DtX] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[posX].disp));
    entityData.position[DtY] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[posY].disp));
    entityData.position[DtZ] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[posZ].disp));

//  printf("\n posx = %f ", entityData.position[DtX]);
//  printf("\n posy = %f ", entityData.position[DtY]);
//  printf("\n posz = %f ", entityData.position[DtZ]);


    // load entity position in geocentric system
//  entityData.position[DtX] = mygeoc[DtX];
//  entityData.position[DtY] = mygeoc[DtY];
//  entityData.position[DtZ] = mygeoc[DtZ];    // altitude

    entityData.velocity[DtX] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[velX].disp));
    entityData.velocity[DtY] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[velY].disp));
    entityData.velocity[DtZ] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[velZ].disp));


    // load acceleration
    entityData.acceleration[DtX] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[accelX].disp));
    entityData.acceleration[DtY] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[accelY].disp));
    entityData.acceleration[DtZ] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[accelZ].disp));

/*
    // perform magic to covert topo orientation to geocentric orientation
    DtTaitBryan topoEuler, geocEuler;
    DtCoordTransform geocToTopo;
    DtGeocToTopoTransform(DtDeg2Rad(  35.699760),
                        DtDeg2Rad(-121.326577), &geocToTopo);
    DtCoordTransform topoToGeoc;
    topoToGeoc.setByInverse(geocToTopo);
    topoEuler = DtTaitBryan(DtDeg2Rad(myHeading),  // heading,pitch,roll
            DtDeg2Rad(myPitch),
            DtDeg2Rad(myRoll));
    topoToGeoc.eulerTrans(topoEuler, &geocEuler);

    // load orientation with geocentric reference
//  entityData.orientation.setPsi(geocEuler.psi());
//  entityData.orientation.setTheta(geocEuler.theta());
//  entityData.orientation.setPhi(geocEuler.phi());
*/

    // load orientation with geocentric reference
    entityData.orientation.setPsi(
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[orientX].disp)));
    entityData.orientation.setTheta(
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[orientY].disp)));
    entityData.orientation.setPhi(
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[orientZ].disp)));


    entityData.rotationalVelocity[DtX] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[rotVelX].disp));
    entityData.rotationalVelocity[DtY] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[rotVelY].disp));
    entityData.rotationalVelocity[DtZ] =
        Gould_to_IEEE_dbl(myHostDataInterface.getDouble(DataMap[rotVelZ].disp));




    // return loaded structure
    return entityData;
};
```

```
//
// HostDataInsertion.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

#ifndef HostDataInsertion_H_
#define HostDataInsertion_H_

#include "vftniuTypes.h"

class HostDataInsertion
{
public:

    HostDataInsertion();
    ~HostDataInsertion();

    void refreshData();

    void putEntityData(entData myEntityData);

private:


};

#endif
```

```
//
// HostDataInsertion.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
This routine uses low level data mapping found in hostDataInterface
to load meaningful groupings of data such as entity data.
*/

#include "stdio.h"

#include "hostDataInsertion.h"
#include "geodCoord.h"
#include "Euler.h"
#include "LibMatrix.h"
#include "topoCoord.h"

#include "hostDataInterface.h"
#include "convType.h"

#include <iostream.h>

hostDataInterface aHostDataInterface;

HostDataInsertion::HostDataInsertion()
{



}

HostDataInsertion::~HostDataInsertion()
{


}



void HostDataInsertion::refreshData()
{
    // update sharred memory from local output buffer
    aHostDataInterface.loadVFTNIUData();

};



// determine entity status
void HostDataInsertion::putEntityData(entData myEntityData)

{

    // define entity types that shall be mapped to host
    char *mySU25 = "1:2:222:2:9:0:0"; // mig
    char *mySU25Missile = "2:1:222:1:13:0:0"; // missile

    int i, validEntity;
    // generic data map
    struct datamap DataMap[numberOfEntityParams];

    validEntity = 0; //set flag off

    // first the entity must be identified/classified so that it can be
    // loaded into the proper memory locations (if applicable)

    // load specific data map in based on entity type


    if (myEntityData.type == mySU25)
    {
        for(i=0; i<numberOfEntityParams; i++)
        DataMap[i] = aHostDataInterface.DomeDataMap[i];
        validEntity = 1; //set flag on
        printf("\n** SU25 mapped to host ");

    }

    if (myEntityData.type == mySU25Missile)
    {
        for(i=0; i<numberOfEntityParams; i++)
        DataMap[i] = aHostDataInterface.DomeMissileMap[i][1];
        validEntity = 1; //set flag on
        printf("\n*** SU25 Missile mapped to host ");
    }

if (validEntity)
{

    // load sharred memory
    aHostDataInterface.putDouble(DataMap[posX].disp,Gould_to_IEEE_dbl(myEntityData.position
[DtX]));
    aHostDataInterface.putDouble(DataMap[posY].disp,Gould_to_IEEE_dbl(myEntityData.position
[DtY]));
    aHostDataInterface.putDouble(DataMap[posZ].disp,Gould_to_IEEE_dbl(myEntityData.position
[DtZ]));

    aHostDataInterface.putDouble(DataMap[velX].disp,Gould_to_IEEE_dbl(myEntityData.velocity
[DtX]));
    aHostDataInterface.putDouble(DataMap[velY].disp,Gould_to_IEEE_dbl(myEntityData.velocity
[DtY]));
    aHostDataInterface.putDouble(DataMap[velZ].disp,Gould_to_IEEE_dbl(myEntityData.velocity
[DtZ]));

    aHostDataInterface.putDouble(DataMap[accelX].disp,Gould_to_IEEE_dbl(myEntityData.accele
ration[DtX]));
    aHostDataInterface.putDouble(DataMap[accelY].disp,Gould_to_IEEE_dbl(myEntityData.accele
ration[DtY]));
    aHostDataInterface.putDouble(DataMap[accelZ].disp,Gould_to_IEEE_dbl(myEntityData.accele
ration[DtZ]));

    aHostDataInterface.putDouble(DataMap[orientX].disp,Gould_to_IEEE_dbl(myEntityData.orien
tation.psi()));
    aHostDataInterface.putDouble(DataMap[orientY].disp,Gould_to_IEEE_dbl(myEntityData.orien
tation.theta()));
    aHostDataInterface.putDouble(DataMap[orientZ].disp,Gould_to_IEEE_dbl(myEntityData.orien
tation.phi()));
```

```
    aHostDataInterface.putDouble(DataMap[rotVelX].disp,Gould_to_IEEE_dbl(myEntityData.rotat
ionalVelocity[DtX]));
    aHostDataInterface.putDouble(DataMap[rotVelY].disp,Gould_to_IEEE_dbl(myEntityData.rotat
ionalVelocity[DtY]));
    aHostDataInterface.putDouble(DataMap[rotVelZ].disp,Gould_to_IEEE_dbl(myEntityData.rotat
ionalVelocity[DtZ]));
}

};
```

```
//
// hostDataInterface.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//
/*
Interfaces to shared memory (shared with HSD process)
*/

#ifndef hostDataInterface_H_
#define hostDataInterface_H_

#include "vftniuTypes.h"
#include <windows.h>

class hostDataInterface
{
    public:

///**** move to private ***
double* doubleAddress;
unsigned char* charAddress;


// map F14 entities coming in from HSD interface (initialized in constructor)
datamap F14DataMap[numberOfEntityParams];
datamap F14MissileDataMap[numberOfEntityParams][maxNumF14Missiles];

// map HLA entities going out to HSD interface (initialized in constructor)
datamap DomeDataMap[numberOfEntityParams];
datamap DomeMissileMap[numberOfEntityParams][maxNumF14Missiles];

    hostDataInterface();
//  ~hostDataInterface();

    // sharred memory members
    // creating processes (vftniu) should (create, map, ... close)
    // attaching processes should ( attatch, map ... close)
    void createSharredMemory();
    void attachSharredMemory();
    void mapSharredMemory();
    void closeSharredMemory();

    // perform double buffering
    void refreshF14Data();   // load from f14/hsd shared mem to local mem buffer
    void loadVFTNIUData();  // load from local mem buffer to sharred mem for export to F14

    // put data into local output buffer
    void putDouble(int displacement, double dvalue);
    void putFloat(int displacement, float fvalue);
    void putInt(int displacement, int ivalue);

    // get data from local input buffer
    double getDouble(int displacement);
    float getFloat(int displacement);
    int getInt(int displacement);



    // FOR TEST PROGRAM TO EMULATE F14
    void putF14Double(int displacement, double dvalue);
    float getF14Float(int displacement);
    void putF14Int(int displacement, int ivalue);

    void emulateHSDIn();    // load from test program to sharred mem to emulate HSD input.
    void obtainHSDOutput(); // load from sharred mem output to local output buffer




    private:

    // for sharred memory
    LPVOID lpMapAddress;
    HANDLE hMapFile, hFile;


    #define inBufferSize 500
    #define outBufferSize 500

    #define inBufferOffset 0
    #define outBufferOffset 250

    unsigned char wstin[inBufferSize];
    unsigned char wstout[outBufferSize];

};

#endif
```

```
//
// hostDataInterface.cpp
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//

/*
Interfaces to shared memory (shared with HSD process)
*/

#include "hostDataInterface.h"   // class header
//#include "stdio.h"
//#include "vftniuTypes.h"

//#include <windows.h>
#include "stdio.h"

int i;
unsigned char* eachAddress;

hostDataInterface::hostDataInterface()
{

// initialize raw data position of entities

F14DataMap[posX].disp = 0;
F14DataMap[posX].type = Double;

F14DataMap[posY].disp = 8;
F14DataMap[posY].type = Double;

F14DataMap[posZ].disp = 16;
F14DataMap[posZ].type = Double;

F14DataMap[velX].disp = 24;
F14DataMap[velX].type = Double;

F14DataMap[velY].disp = 32;
F14DataMap[velY].type = Double;

F14DataMap[velZ].disp = 40;
F14DataMap[velZ].type = Double;

F14DataMap[accelX].disp = 48;
F14DataMap[accelX].type = Double;

F14DataMap[accelY].disp = 56;
F14DataMap[accelY].type = Double;

F14DataMap[accelZ].disp = 64;
F14DataMap[accelZ].type = Double;

F14DataMap[orientX].disp = 72;
F14DataMap[orientX].type = Double;

F14DataMap[orientY].disp = 80;
F14DataMap[orientY].type = Double;

F14DataMap[orientZ].disp = 88;
F14DataMap[orientZ].type = Double;
```

```
F14DataMap[rotVelX].disp = 96;
F14DataMap[rotVelX].type = Double;

F14DataMap[rotVelY].disp = 104;
F14DataMap[rotVelY].type = Double;

F14DataMap[rotVelZ].disp = 112;
F14DataMap[rotVelZ].type = Double;

F14DataMap[status].disp = 120;
F14DataMap[status].type = Int;

int missile1Disp = 124;

F14MissileDataMap[posX][1].disp = 0 + missile1Disp;
F14MissileDataMap[posX][1].type = Double;

F14MissileDataMap[posY][1].disp = 8 + missile1Disp;
F14MissileDataMap[posY][1].type = Double;

F14MissileDataMap[posZ][1].disp = 16 + missile1Disp;
F14MissileDataMap[posZ][1].type = Double;

F14MissileDataMap[velX][1].disp = 24 + missile1Disp;
F14MissileDataMap[velX][1].type = Double;

F14MissileDataMap[velY][1].disp = 32 + missile1Disp;
F14MissileDataMap[velY][1].type = Double;

F14MissileDataMap[velZ][1].disp = 40 + missile1Disp;
F14MissileDataMap[velZ][1].type = Double;

F14MissileDataMap[accelX][1].disp = 48 + missile1Disp;
F14MissileDataMap[accelX][1].type = Double;

F14MissileDataMap[accelY][1].disp = 56 + missile1Disp;
F14MissileDataMap[accelY][1].type = Double;

F14MissileDataMap[accelZ][1].disp = 64 + missile1Disp;
F14MissileDataMap[accelZ][1].type = Double;

F14MissileDataMap[orientX][1].disp = 72 + missile1Disp;
F14MissileDataMap[orientX][1].type = Double;

F14MissileDataMap[orientY][1].disp = 80 + missile1Disp;
F14MissileDataMap[orientY][1].type = Double;

F14MissileDataMap[orientZ][1].disp = 88 + missile1Disp;
F14MissileDataMap[orientZ][1].type = Double;

F14MissileDataMap[rotVelX][1].disp = 96 + missile1Disp;
F14MissileDataMap[rotVelX][1].type = Double;

F14MissileDataMap[rotVelY][1].disp = 104 + missile1Disp;
F14MissileDataMap[rotVelY][1].type = Double;

F14MissileDataMap[rotVelZ][1].disp = 112 + missile1Disp;
F14MissileDataMap[rotVelZ][1].type = Double;
```

```
F14MissileDataMap[status][1].disp = 120 + missile1Disp;
F14MissileDataMap[status][1].type = Int;


int domeDisp = missile1Disp + 124;


DomeDataMap[posX].disp = 0 + domeDisp;
DomeDataMap[posX].type = Double;

DomeDataMap[posY].disp = 8 + domeDisp;
DomeDataMap[posY].type = Double;

DomeDataMap[posZ].disp = 16 + domeDisp;
DomeDataMap[posZ].type = Double;

DomeDataMap[velX].disp = 24 + domeDisp;
DomeDataMap[velX].type = Double;

DomeDataMap[velY].disp = 32 + domeDisp;
DomeDataMap[velY].type = Double;

DomeDataMap[velZ].disp = 40 + domeDisp;
DomeDataMap[velZ].type = Double;

DomeDataMap[accelX].disp = 48 + domeDisp;
DomeDataMap[accelX].type = Double;

DomeDataMap[accelY].disp = 56 + domeDisp;
DomeDataMap[accelY].type = Double;

DomeDataMap[accelZ].disp = 64 + domeDisp;
DomeDataMap[accelZ].type = Double;

DomeDataMap[orientX].disp = 72 + domeDisp;
DomeDataMap[orientX].type = Double;

DomeDataMap[orientY].disp = 80 + domeDisp;
DomeDataMap[orientY].type = Double;

DomeDataMap[orientZ].disp = 88 + domeDisp;
DomeDataMap[orientZ].type = Double;

DomeDataMap[rotVelX].disp = 96 + domeDisp;
DomeDataMap[rotVelX].type = Double;

DomeDataMap[rotVelY].disp = 104 + domeDisp;
DomeDataMap[rotVelY].type = Double;

DomeDataMap[rotVelZ].disp = 112 + domeDisp;
DomeDataMap[rotVelZ].type = Double;


int domeMissile1Disp = domeDisp + 120;


DomeMissileMap[posX][1].disp = 0 + domeMissile1Disp;
DomeMissileMap[posX][1].type = Double;

DomeMissileMap[posY][1].disp = 8 + domeMissile1Disp;
```

```
DomeMissileMap[posY][1].type = Double;

DomeMissileMap[posZ][1].disp = 16 + domeMissile1Disp;
DomeMissileMap[posZ][1].type = Double;

DomeMissileMap[velX][1].disp = 24 + domeMissile1Disp;
DomeMissileMap[velX][1].type = Double;

DomeMissileMap[velY][1].disp = 32 + domeMissile1Disp;
DomeMissileMap[velY][1].type = Double;

DomeMissileMap[velZ][1].disp = 40 + domeMissile1Disp;
DomeMissileMap[velZ][1].type = Double;

DomeMissileMap[accelX][1].disp = 48 + domeMissile1Disp;
DomeMissileMap[accelX][1].type = Double;

DomeMissileMap[accelY][1].disp = 56 + domeMissile1Disp;
DomeMissileMap[accelY][1].type = Double;

DomeMissileMap[accelZ][1].disp = 64 + domeMissile1Disp;
DomeMissileMap[accelZ][1].type = Double;

DomeMissileMap[orientX][1].disp = 72 + domeMissile1Disp;
DomeMissileMap[orientX][1].type = Double;

DomeMissileMap[orientY][1].disp = 80 + domeMissile1Disp;
DomeMissileMap[orientY][1].type = Double;

DomeMissileMap[orientZ][1].disp = 88 + domeMissile1Disp;
DomeMissileMap[orientZ][1].type = Double;

DomeMissileMap[rotVelX][1].disp = 96 + domeMissile1Disp;
DomeMissileMap[rotVelX][1].type = Double;

DomeMissileMap[rotVelY][1].disp = 104 + domeMissile1Disp;
DomeMissileMap[rotVelY][1].type = Double;

DomeMissileMap[rotVelZ][1].disp = 112 + domeMissile1Disp;
DomeMissileMap[rotVelZ][1].type = Double;



}


void hostDataInterface::createSharredMemory()
{
    //create file mappinf object
    hMapFile = CreateFileMapping(hFile,     // Current file handle.
        NULL,                               // Default security.
        PAGE_READWRITE,                     // Read/write permission.
        0,                                  // Max. object size.
        inBufferSize + outBufferSize,       // Size of hFile.
        "MyFileMappingObject");             // Name of mapping object.

    if (hMapFile == NULL)
        printf(" \n ERROR Could not create file-mapping object " );
    else
        printf(" \n created file-mapping object " );
```

```
}

void hostDataInterface::attachSharredMemory()
{

    // open shared mem created by another process
    hMapFile = OpenFileMapping(FILE_MAP_ALL_ACCESS, // Read/write permission.
        FALSE,                               // Do not inherit the name
        "MyFileMappingObject");              // of the mapping object.

    if (hMapFile == NULL) {
    printf("  Could not open file-mapping object " );
    }

}

void hostDataInterface::mapSharredMemory()
{


    // create view of shared mem
    lpMapAddress = MapViewOfFile(hMapFile, // Handle to mapping object.
      FILE_MAP_ALL_ACCESS,                // Read/write permission
      0,                                  // Max. object size.
      0,                                  // Size of hFile.
      0);                                 // Map entire file.

    if (lpMapAddress == NULL)
        printf(" \n ERROR Could not map view of file " );
    else
        printf(" \n map view of file " );


    // convert pointer to unsigned long pointer
    doubleAddress = (double*) lpMapAddress;
    charAddress = (unsigned char*) lpMapAddress;

//  printf(" \n initial address is %x  " ,myaddress);
//  printf(" \n initial value is %d  " ,*((LPDWORD) myaddress));


}

void hostDataInterface::closeSharredMemory()
{
    // clean up shared memory usage  (each process must do this)
    if (!UnmapViewOfFile(lpMapAddress))
        printf("\n ERROR  Could not unmap view of file.");
    else
       printf("\n unmap view of file and close");

    CloseHandle(hMapFile);  // close mapped file.


}


void hostDataInterface::refreshF14Data()
// refresh local input buffer from f14/hsd shared mem for use by VFTNIU
```

```
{
    // get base address;
    eachAddress = charAddress + inBufferOffset;

    for (i=0;i<inBufferSize;i++)
        wstin[i] = *eachAddress++;     // from shared memory
}


void hostDataInterface::loadVFTNIUData()
// load local output buffer into sharred mem for export to F14

{

    // get base address;
    eachAddress = charAddress + outBufferOffset;

    for (i=0;i<outBufferSize;i++)
        *eachAddress++ = wstout[i];     // to shared memory

}


void hostDataInterface::emulateHSDIn()
// ** test routine **
// load sharred memory with local input buffer.
// This is used by test program to emulate the HSD input transfer

{
    // get base address;
    eachAddress = charAddress + inBufferOffset;

    for (i=0;i<inBufferSize;i++)
        *eachAddress++ = wstin[i];     // to shared memory
}



void hostDataInterface::obtainHSDOutput()
// ** test routine **
// load local memory with sharred memory output buffer.
// This is used by test program to access data that is loaded
// for HSD output.

{
    // get base address;
    eachAddress = charAddress + outBufferOffset;

    for (i=0;i<outBufferSize;i++)
        wstout[i] = *eachAddress++;     // from shared memory
}



double hostDataInterface::getDouble(int displacement)
// get double from input buffer

{
    union {
        unsigned char word [8];
        double mydouble;
        //unsigned short myint [4];
    };

    int j;
    j = 0;
    mydouble = 0;

//  printf("\n put double, disp = %d \n", displacement);

    for (j = 0; j <= 7; j++) {
        word [j] = wstin [displacement + (7-j)];
//      printf(" wstin[%d] = %x ", displacement+j,wstin[j]);

    }

//    printf("\n get disp = %d word[%d] = %x ", displacement,j,word[j]);

    return mydouble;
}


void hostDataInterface::putDouble(int displacement, double dvalue)
// put double to output buffer

{
    union
    {
        unsigned char word [8];
        double myDouble;
    };

    myDouble = dvalue;

//  printf("\n put double, disp = %d \n", displacement);

    int j;
    for (j = 0; j <= 7; j++)
    {
        wstout [displacement + (j)] = word [7-j];
//      printf(" wstin[%d] = %x ", displacement+j,wstin[j]);
    }
}


float hostDataInterface::getFloat(int displacement)
{
    union
    {
        unsigned char word [4];
        float myfloat;
    };

    int j;
    for (j = 0; j <= 3; j++)
        word [j] = wstin [displacement + (3-j)];

    return myfloat;
}
```

```
void hostDataInterface::putFloat(int displacement, float fvalue)
{
}


int hostDataInterface::getInt(int displacement)
{
    union {
        unsigned char word [4];
        long myInt;
    };

    int j;
    for (j = 0; j <= 3; j++)
    {
        word [j] = wstin [displacement + (3-j)];
//      printf(" wstin[%d] = %x ", displacement+j,wstin[displacement+j]);
    }

    return myInt;
}


void hostDataInterface::putInt(int displacement, int ivalue)
{
    union
    {
        unsigned char word [4];
        long myInt;
    };

    myInt = ivalue;


    int j;
    for (j = 0; j <= 3; j++)
    {
        wstout [displacement + (j)] = word [3-j];
//      printf(" wstout[%d] = %x ", displacement+j,wstout[j]);
    }
}



void hostDataInterface::putF14Double(int displacement, double dvalue)
// for testing (emulated HSD read operation from WST)
{

    union
    {
        unsigned char word [8];
        double myDouble;
    };

    myDouble = dvalue;

//  printf("\n put double, disp = %d \n", displacement);
```

```
    int j;
    for (j = 0; j <= 7; j++)
    {
        wstin [displacement + (j)] = word [7-j];
//      printf(" wstin[%d] = %x ", displacement+j,wstin[j]);
    }
}


void hostDataInterface::putF14Int(int displacement, int ivalue)
// for testing (load input buffer to emulated HSD read operation from WST)
{

    union
    {
        unsigned char word [4];
        long myInt;
    };

    myInt = ivalue;

//  printf("\n putF14Int = %d ",myInt);

    int j;
    for (j = 0; j <= 3; j++)
    {
        wstin [displacement + (j)] = word [3-j];
//      printf(" wstin[%d] = %x ", displacement+j,wstin[displacement+j]);
    }
}


float hostDataInterface::getF14Float(int displacement)
// ** for testing **
// get a float from HSD output buffer
// for testing what has been loaded into HSD output area
{
    union
    {
        unsigned char word [4];
        float myfloat;
    };

    //displacement = disp;
    int j;

    for (j = 0; j <= 3; j++)
        word [j] = wstout [displacement + (3-j)];

    return myfloat;
}
```

convTypes.h

```
//
// convTypes.h
// Randy Lang, Dual Incorporated 6/23/1999
// SBIR TOPIC N96-053
//


/***********************************************************************/
/* Convert an IEEE double to a Gould double                           */
/***********************************************************************/
double IEEE_to_Gould_dbl(double dbl);
double testIEEE_to_Gould_dbl(double myDouble);  // prints debug values

void printGouldDouble(double gouldDouble);  // for printing hex value

/***********************************************************************/
/* Convert an IEEE float to a Gould float                             */
/***********************************************************************/
float IEEE_to_Gould_flt(float flt);
float testIEEE_to_Gould_flt(float flt);



/***********************************************************************/
/* Convert a Gould float to a IEEE standard float                     */
/***********************************************************************/
float Gould_to_IEEE_flt(float flt);
float testGould_to_IEEE_flt(float flt);


/***********************************************************************/
/* Convert a Gould double to a IEEE standard double                   */
/***********************************************************************/
double Gould_to_IEEE_dbl(double dbl);
double testGould_to_IEEE_dbl(double myDouble);  // prints debug values
```

```
/*********************************************************************
File:   convType.c
Author Roger Schwabe /modified by Randy Lang
    march 1992/June 99

This file performs floating point conversion between Gould and IEEE
formats.

Examples:     Value     IEEE        Gould
                1.0     0x3f800000  0x41100000
                1.5     0x3fc00000  0x41180000
               -1.5     0xbfc00000  0xbee80000
----------------------------------------------------
Test Values     160                 0x429FFFFF
                16000

functions:
    IEEE_to_Gould_flt;
    IEEE_to_Gould_dbl;
    Gould_to_IEEE_flt;
    Gould_to_IEEE_dbl;
*********************************************************************/

#include "convType.h"

#include <iostream.h>
#include <stdio.h>
#include <math.h>

typedef struct{
    union {
       float f;
           int   i;  // causing trouble?
       struct {
        unsigned frac:23;
        unsigned expon:8;
        unsigned sign:1;

          } word;
       struct {unsigned char data[4]; } word1;
    } floater;
} IEEE_flt;

typedef struct {

    union {
        float f;
            int i; // same as i above
        struct {
            unsigned frac:24;
            unsigned expon:7;
            unsigned sign:1;
            } word;
        struct {unsigned char data[4]; } word1;
    } floater;
} Gould_flt;

typedef struct{
    union {
        double f;
```

```
                struct {
                        int i2;
                        int i1;
//                        int i3;
//                        int i4;
                       } i;
        struct {
            unsigned frac2:32;
            unsigned frac1:20;
            unsigned expon:11;
            unsigned sign:1;
            } word;
        struct {unsigned char data[8]; } word1;
    } floater;
} IEEE_dbl;

typedef struct {
    union {
        double f;
                struct {
                       int i2;       // 4 bytes
                       int i1;       // 4 bytes
                        } i;
        struct {
            unsigned frac2:32;
            unsigned frac1:24;
            unsigned expon:7;
            unsigned sign:1;
            } word;
        struct {unsigned char data[8]; } word1;
    } floater;
} Gould_dbl;




/***************************************************************************/
/* Convert an IEEE double to a Gould double                                */
/***************************************************************************/


double IEEE_to_Gould_dbl(double dbl)
{

int temp;   // 4 bytes
unsigned long shift, shift_bits;
int float_exp;

Gould_dbl Gould_double;
IEEE_dbl IEEE_double;

// move bits into IEEE format
IEEE_double.floater.f = dbl;


// initialize
temp = 0;
Gould_double.floater.f = 0.0;
Gould_double.floater.i.i1 = 0;
Gould_double.floater.i.i2 = 0;


if(IEEE_double.floater.f != 0){


    // take out bias from exponent
    float_exp = IEEE_double.floater.word.expon - 0x3ff; // changed to correct bias value
    //printf("\n float_exp = %d \n",float_exp);

    // get IEEE fract 1 (20 bits)
    temp = IEEE_double.floater.word.frac1;

    // move 20 bits to 24 bit field (this will need adjustment)
    Gould_double.floater.word.frac1 = IEEE_double.floater.word.frac1;

    // fract2 can be moved in with no adjustments
    Gould_double.floater.word.frac2 = IEEE_double.floater.word.frac2;


    // add 1.0 to account for implied 1.xx in ieee value (not implied in gould)
    Gould_double.floater.i.i1 += 0x100000;


    // determine shift amount
    // shift is necessary because of diff in exponents (Right shift 0-3)
    // and left shift of 4 because of 24 to 20 bit diff in frac1
    if ((float_exp % 4) == 0)  shift = 0;
    else if ((float_exp % 4) < 0) shift = 4 - abs(float_exp % 4);
    else shift = (float_exp % 4);

    shift_bits = 0;

    // save certain bits depending on amount shifted
    switch (shift) {
            case 1: shift_bits = Gould_double.floater.word.frac2 & 0x80000000;
                    break;
            case 2: shift_bits = Gould_double.floater.word.frac2 & 0xC0000000;
                    break;
            case 3: shift_bits = Gould_double.floater.word.frac2 & 0xE0000000;
                    break;
            case 4: shift_bits = Gould_double.floater.word.frac2 & 0xF0000000;
                    break;
    }

    // shift fract 1
    Gould_double.floater.i.i1 <<= shift;

    // shift fract 2
    Gould_double.floater.word.frac2 <<= shift;

    // right justify shift bits
    shift_bits >>= 32 - shift;

    // inclusive or (turn shift bits on) (saved from frac2)
    Gould_double.floater.word.frac1 |= shift_bits;

    // convert exponent to gould
    if( (float_exp < 0) & (float_exp%4 != 0))
        Gould_double.floater.word.expon =(float_exp/4)+ 0x40;
    else
        Gould_double.floater.word.expon =(float_exp/4) + 1 + 0x40;


    // convert if negative
    if (IEEE_double.floater.word.sign) {
        if (IEEE_double.floater.i.i2 == 0) {
            Gould_double.floater.i.i2 = 0;
            Gould_double.floater.i.i1 = -Gould_double.floater.i.i1;
        }
        else{
            Gould_double.floater.i.i2 = -(Gould_double.floater.i.i2);
            Gould_double.floater.i.i1 = ~(Gould_double.floater.i.i1);
         } // if negative number
    }
}
else
    // if 0.0
    Gould_double.floater.f = 0;

    /* useful for debugging
    printf("\n w7=%2x w6=%2x w5=%2x w4=%2x w3=%2x w2=%2x w1=%2x  w0=%2x \n\n",
        Gould_double.floater.word1.data[7],
        Gould_double.floater.word1.data[6],
        Gould_double.floater.word1.data[5],
        Gould_double.floater.word1.data[4],
        Gould_double.floater.word1.data[3],
        Gould_double.floater.word1.data[2],
        Gould_double.floater.word1.data[1],
        Gould_double.floater.word1.data[0]
        );
    */

return(Gould_double.floater.f);

}



/************************************************************************/
/* Convert an IEEE float to a Gould float                               */
/************************************************************************/

float IEEE_to_Gould_flt(float flt)
{
Gould_flt Gould_float;
IEEE_flt IEEE_float;
int temp = 45;
int float_exp;


   IEEE_float.floater.i = 0;
   Gould_float.floater.i = 0;

   // load float into union
   IEEE_float.floater.f = flt;


   if(IEEE_float.floater.i != 0){
    // take bias out of exponent
       float_exp = IEEE_float.floater.word.expon - 0x7e;

      // extraction of fraction
      temp = IEEE_float.floater.word.frac;
      Gould_float.floater.word.frac = temp + 0x800000;


      if (float_exp%4 != 0) {
         Gould_float.floater.word.frac >>= (4-float_exp%4);

         Gould_float.floater.word.expon = float_exp/4 + 0x41;

      }
      else
         Gould_float.floater.word.expon = float_exp/4 + 0x40;

      if (IEEE_float.floater.word.sign)
     Gould_float.floater.i = -Gould_float.floater.i;
   }
   else {
      Gould_float.floater.i = 0;
   }

  return (Gould_float.floater.f);

}


/***************************************************************************/
/* Convert a Gould float to a IEEE standard float                         */
/***************************************************************************/

float Gould_to_IEEE_flt(float flt)
{
Gould_flt temp;
int float_exp, float_man, sign;
IEEE_flt IEEE;
Gould_flt Gould;

  Gould.floater.f = flt;

  IEEE.floater.i = 0;
  if(Gould.floater.i != 0){

     temp.floater.i = Gould.floater.i;
     if (temp.floater.i < 0) {
        temp.floater.i = -temp.floater.i;
        sign = 1;
     }
     float_exp = (temp.floater.word.expon - 0x40) << 2;
     float_man = temp.floater.word.frac;

     while (( (float_man & 0x800000 ) == 0) && (float_man !=0)) {
         float_man <<= 1;
         float_exp -= 1;
      }
     float_exp += 0x7e;
     float_man &= 0x007fffff;
    if (Gould.floater.word.sign)
       IEEE.floater.word.sign = 1;
    else
       IEEE.floater.word.sign = 0;
    IEEE.floater.word.expon = float_exp;
    IEEE.floater.word.frac = float_man;
  }
  else {
    IEEE.floater.i = 0;
  }

  return(IEEE.floater.f);

} /* Gould_to_IEEE_flt */




/*************************************************************************/
/* Convert a Gould double to a IEEE standard double                      */
/*************************************************************************/

double Gould_to_IEEE_dbl( double dbl )
{

Gould_dbl Gould,temp;
IEEE_dbl IEEE;

long float_exp, float_man;
int shift, shift_bits;

// load gould float variable
Gould.floater.f = dbl;

IEEE.floater.f = 0;


if(Gould.floater.f != 0){

temp.floater.f = Gould.floater.f;

// if negative
if (temp.floater.word.sign) {
    if (temp.floater.i.i2 == 0) {
        temp.floater.i.i2 = 0;
        temp.floater.i.i1 = -temp.floater.i.i1;
    }
    else{
        temp.floater.i.i2 = -(temp.floater.i.i2);
        temp.floater.i.i1 = ~(temp.floater.i.i1);
    } // if negative number
}


float_exp = (temp.floater.word.expon - 0x40) << 2;


float_man = temp.floater.word.frac1;
```

```
shift = 0;

// while 25th bit in mantissa is not set and mantissa not 0 , shift bits left
while (( (float_man & 0x01000000) == 0) && (float_man !=0)) {
    float_man <<= 1;
    float_exp -= 1;
    shift += 1;
}

//     float_exp += 0x3fe;
float_exp += 0x3ff;  // updated by rsl 6/14/99

shift_bits = 0;
switch (shift) {
    case 1: shift_bits = temp.floater.word.frac1 & 0x7;
        break;
    case 2: shift_bits = temp.floater.word.frac1 & 0x3;
        break;
    case 3: shift_bits = temp.floater.word.frac1 & 0x1;
        break;
    case 4: break;
}

shift = 4-shift;
shift_bits <<= (32-shift);
float_man >>= 4;
float_man &= 0x000fffff;


IEEE.floater.word.frac2 = temp.floater.word.frac2;
IEEE.floater.word.frac2 >>= shift;

// add shifted bits
IEEE.floater.word.frac2 |= shift_bits;


// if negative
if (Gould.floater.word.sign)
    IEEE.floater.word.sign = 1;
else
    IEEE.floater.word.sign = 0;

IEEE.floater.word.expon = float_exp;
IEEE.floater.word.frac1 = float_man;

}
else {
    IEEE.floater.f = 0;
}

return (IEEE.floater.f);

} /* Gould_to_IEEE_dbl */
```

```
// test routines


void printGouldDouble(double gouldDouble)  // for printing hex value of real
{

    union {
        double dl;
        struct {
        long lo2;
        long lo1;
                }lo;
    } myLong;

    myLong.dl = gouldDouble;
    printf("%08X %08x", myLong.lo.lo1, myLong.lo.lo2);

}


void printGouldfloat(float gouldFloat)  // for printing hex value of real
{
    union {
        float fl;
        long lo;
    }myGouldFloat;

    myGouldFloat.fl = gouldFloat;
    printf("%08X", myGouldFloat.lo);

}

  double testIEEE_to_Gould_dbl(double myDouble)
  {
  printf("\n Input ieee double = %f  out gould long hex: ", myDouble);

  double myOutDouble;
  myOutDouble = IEEE_to_Gould_dbl(myDouble);

    printGouldDouble(myOutDouble);

    return myOutDouble;
  }


double testGould_to_IEEE_dbl(double myDouble)
{
    printf("\n Input gould double = ");
    printGouldDouble(myDouble);

    double myOutDouble;
    myOutDouble = Gould_to_IEEE_dbl(myDouble);

    printf(" output ieee double = %f", myOutDouble);

    return myOutDouble;
}
```

```
float testGould_to_IEEE_flt(float flt)
{
    printf("\n Input gould float = ");
    printGouldfloat(flt);

    float myOutFloat;
    myOutFloat = Gould_to_IEEE_flt(flt);

    printf(" output ieee float = %f", myOutFloat);

    return myOutFloat;
}



float testIEEE_to_Gould_flt(float flt)
{
    printf("\n Input ieee float = %f  out gould long hex: ", flt);

    float myOutfloat;
    myOutfloat = IEEE_to_Gould_flt(flt);

    printGouldfloat(myOutfloat);

    return myOutfloat;
}
```

# APPENDIX F

# VERSATILE FLIGHT TRAINER NETWORK INTERFACE UNIT USERS GUIDE

# VFTNIU USERS GUIDE

**General**: The Versatile Flight Network Interface Unit (VFTNIU) is a product developed by Dual Incorporated that provides a quick, low cost, alternative for flight simulators seeking HLA compliance. The VFTNIU consists of a windows NT computer and software products that provides the gateway to an HLA network. The unique characteristic of this gateway system is that it uses Mak's VR-Link software to interface to the HLA network. Dual Incorporated adds middleware that provides data conversion and flexible data mapping techniques. This results in a gateway system that can seemlessly evolve with HLA by leveraging the VR-Link API.

**The VFTNIU Demonstration Environment:** Dual Incorporated has delivered the VFTNIU as part of a demostration package. This includes three Intergraph computers refered to as the STEALTH, the VFTNIU, and the HLA_SIM. The NT workstations are networked together using ethernet.

The STEALTH computer houses MAK's Stealth viewer and Logger. This is also where the RTIEXEC and MAK's license manager is launched.

The VFTNIU computer houses the VFTNIU software. This computer facilitates the gateway to the HLA network. Currently there is no physical connection implemented to a host computer. The interface is defined as a sharred memory area on the VFTNIU implemented with windows file mapping. This allows for any type of physical interface to be mapped into this data area from an independant interface task.

The HLA_SIM computer provides a test platform for an HLA compliant simulator. Dual has developed a simple flight simulator that flies with a joystick and shoots missiles. The program uses MAK's VR-Link as the API to access the network. This program will be referred to below as the Dual Sim. The Dual Sim may also be ran on the VFTNIU computer as described below.

**Starting the VFTNIU:**
1) On the STEALTH computer, start the MAK license manager. This can be done by double clicking the icon on the desktop or starting "D:\Logger\flexlm\runLm.bat". This license manager accomodates all Mak products on the network including: the STEALTH/LOGGER, The VFTNIU, and Dual's HLA SIM. The liscense will allow for a maximum of two VR-Link products to run simultaneously along with the Stealth/Logger. Example programs (provided as executables from Mak) may run without a license.

2) On the STEALTH computer, start the RTIEXEC. This can be done by double clicking on the desktop icon or by running "C:\Program Files\DMSO\RTI1.3v4\bin\WIN32\rtiexec.exe". This program was downloaded from DMSO to support RTIEXEC version 1.3v4.

3) On the STEALTH computer, start the STEALTH. This can be done by double clicking on the desktop icon or by running "C:\Program Files\MAK Technologies\Stealth\StealthRPR.exe". The PC Stealth View window must be colapsed by dragging the bottom border up. The Mak Stealth window must be minimized. See the help pull down for a Stealth on-line users manual.

4) On the VFTNIU computer, start the VFTNIU. This can be done by double clicking on the desktop icon or by running "C:\Randy\vftniuWorkArea\vftniu\debug\vftniu.exe". The VFTNIU receives entity information through sharred memory and transfers it to the HLA network. Conversely, particular entity information is received from the HLA network and is loaded into sharred memory. To test the sharred memory to HLA functionality, the F14 Emulator program must be ran. To test the HLA to sharred memory route the Dual Simulator must be ran.

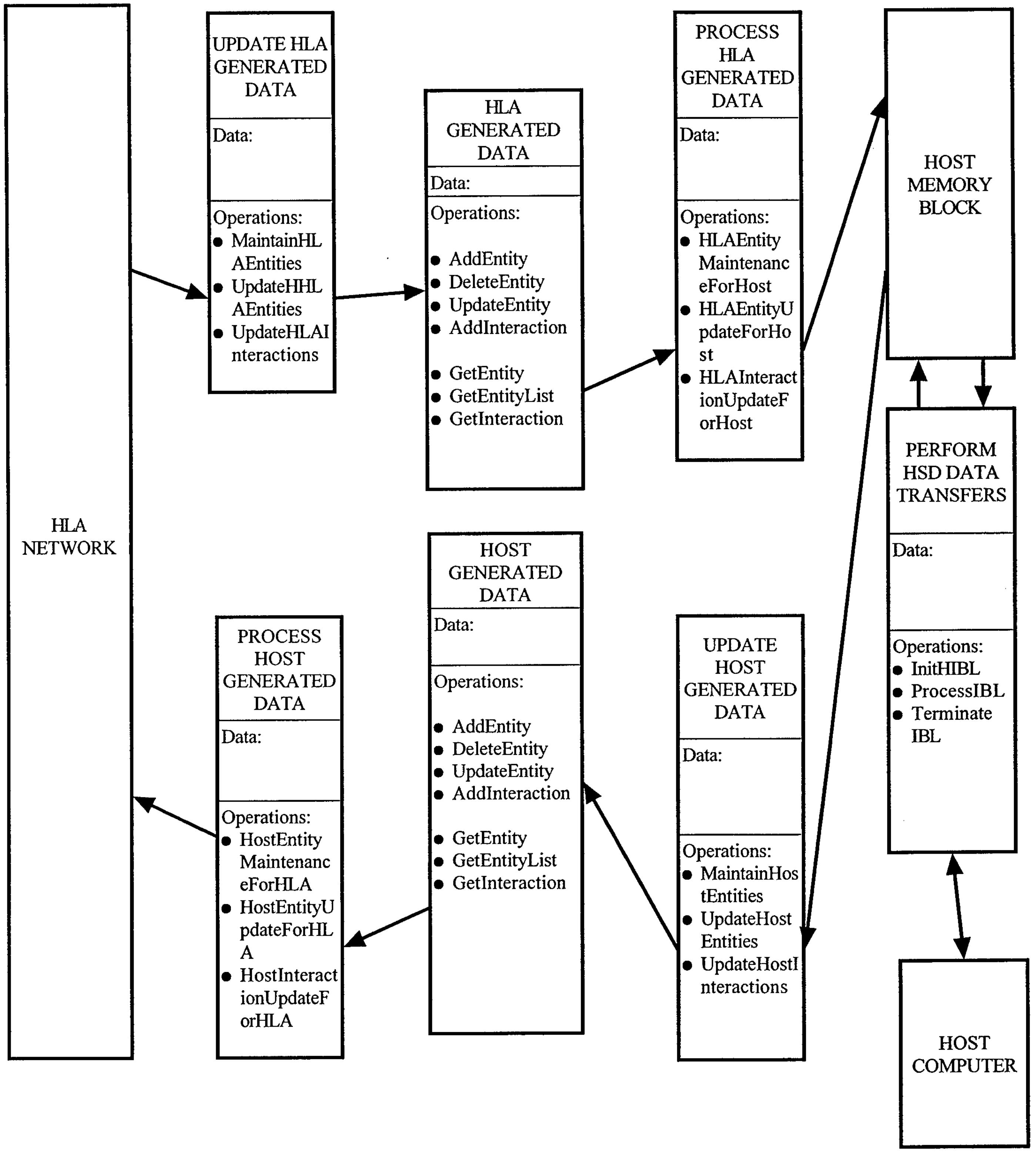
**Demonstrating the VFTNIU:**

**F14 Emulator:** Start the VFTNIU as decribed above. On the VFTNIU computer, start the F14 Emulator. This can be done by double clicking on the desktop icon or by running C:\Randy\vftniuWorkArea\F14Emulator\debug\F14Emulator.exe". This program simulates simulates a static flight and puts data (in Encore format) into sharred memory as would the F14 Tomcat using a block memory transfer such as HSD or Ethernet. In addition this program transmits entity information to the HLA network as a way of verifying VFTNIU transmissions. This test data will appear on the stealth viewer as a helicopter. The VFTNIU picks up the data from sharred memory, converts it, and transfers it to the HLA network. This VFTNIU transmitted data appears as an F18. The Stealth is used to verify that the visual representations of the VFTNIU transmission corresponds to the visual representation that the F14 Emulator transmitted. Note that the smoothing function on the stealth viewer may need to be set to "none" in order to be able to see entities properly.

**Dual Simulator:** Start the VFTNIU as decribed above. On the VFTNIU computer, start the Dual Simulator. This can be done by double clicking on the desktop icon or by running C:\Randy\vftniuWorkArea\dualSim\debug\dualSim.exe". The Dual Simulator is controlled with the microsoft joystick. Speed is controllable with the throttle slider. The two buttons on the upper left control reset and freeze functions. The trigger fires a missile that will seek the nearest entity and detonate in proximaty to the target. The Dual Sim can be "flown" by attaching to the entity with the Stealth viewer in mimic mode. Entity location and fire/detonation interactions will be trasmitted to the HLA network. The VFTNIU will pickup these entities and interactions from the HLA network and load them into sharred memory in Encore format. Targets may be initiated into the HLA network by double clicking on the icon on the VFTNIU computers desktop. These targets can be shot down by the DualSim.

# APPENDIX G

# HIGH LEVEL LOGICAL MODEL


HLA NETWORK
UPDATE HLA GENERATED DATA
Data:
Operations:
● MaintainHLAEntities
● UpdateHHLAEntities
● UpdateHLAInteractions
HLA GENERATED DATA
Data:
Operations:
● AddEntity
● DeleteEntity
● UpdateEntity
● AddInteraction
● GetEntity
● GetEntityList
● GetInteraction
PROCESS HLA GENERATED DATA
Data:
Operations:
● HLAEntityMaintenanceForHost
● HLAEntityUpdateForHost
● HLAInteractionUpdateForHost
HOST MEMORY BLOCK
PERFORM HSD DATA TRANSFERS
Data:
Operations:
● InitHIBL
● ProcessIBL
● Terminate IBL
HOST COMPUTER
PROCESS HOST GENERATED DATA
Data:
Operations:
● HostEntityMaintenanceForHLA
● HostEntityUpdateForHLA
● HostInteractionUpdateForHLA
HOST GENERATED DATA
Data:
Operations:
● AddEntity
● DeleteEntity
● UpdateEntity
● AddInteraction
● GetEntity
● GetEntityList
● GetInteraction
UPDATE HOST GENERATED DATA
Data:
Operations:
● MaintainHostEntities
● UpdateHost Entities
● UpdateHostInteractions

# APPENDIX H

# SIMULATION OBJECT MODEL

Object Model Identification Table

| Category | Information | |
|---|---|---|
| Name | Tomcat SOM | |
| Version | 1.0 | |
| Date | 06/14/1999 | |
| Purpose | To identify current F14 Tocat HLA attributes and interactions | |
| Application Domain | Real time, platform level simulations | |
| Sponsor | NAVAIR, PMA-2051BF | |
| POC (Title, First, Last) | Mr | Randall |
| POC Organization | Dual Incorporated | |
| POC Telephone | 4072828678 | |
| POC Email | rlang@dualinc.com | |

| |
|---|
| |
| |
| |
| |
| |
| |
| Lang |
| |
| |
| |

Object Class Structure Table

| Class1 | Class2 | Class3 |
|---|---|---|
| BaseEntity (S) | PhysicalEntity [1] (PS) | MilitaryEntity (S) |
| | | CivilPlatform (PS) |
| | | Civilian (PS) |

| Class4 | Class5 |
|---|---|
| MilitaryPlatformEntity (PS) | MilitaryAirLandPlatform (PS) |
| | MilitaryAmphibiousPlatform (P |
| | MilitaryLandPlatform (PS) |
| | MilitarySpacePlatform (PS) |
| | MilitarySeaSurfacePlatform (P |
| | MilitarySubmersiblePlatform (P |
| | MilitaryMultiDomainPlatform (P |
| MunitionEntity (PS) | |
| Soldier (PS) | |
| CivilAirLandPlatform (PS) | |
| CivilAmphibiousPlatform (PS) | |
| CivilLandPlatform (PS) | |
| CivilSpacePlatform (PS) | |
| CivilSeaSurfacePlatform (PS) | |
| CivilSubmersiblePlatform (PS) | |
| CivilMultiDomainPlatform (PS) | |
| | |

| Interaction1 |
|---|
| MunitionDetonation (IR) |
| WeaponFire (IR) |

| Object | Attribute | Datatype |
|---|---|---|
| BaseEntity | AccelerationVector | AccelerationStruct |
| | AngularVelocityVector | AngVelocityStruct |
| | DRAlgorithm | DRAlgorithmEnum |
| | EntityType | EntityTypeStruct |
| | EntityID | EntityIDStruct |
| | IsFrozen | boolean |
| | Orientation | OrientationStruct |
| | Position | PositionStruct |
| | VelocityVector | VelocityStruct |
| MilitaryEntity | AlternateEntityType | EntityTypeStruct |
| | CamouflageType | CamouflageEnum |
| | FirePowerDisabled | boolean |
| | ForceID | ForceIdEnum |
| | IsConcealed | boolean |
| MilitaryPlatformEntity | AfterburnerOn | boolean |
| | HasAmmunitionSupplyCap | boolean |
| | LauncherRaised | boolean |
| MunitionEntity | LauncherFlashPresent | boolean |
| PhysicalEntity [1] | ArticulatedParametersArray | ArticulatedParameterStruct |
| | DamageState | DamageStateEnum |
| | EngineSmokeOn | boolean |
| | FlamesPresent | boolean |
| | HasFuelSupplyCap | boolean |
| | HasRecoveryCap | boolean |
| | HasRepairCap | boolean |
| | HatchState | HatchStateEnum |
| | Immobilized | boolean |
| | LifeformState | LifeformStateEnum |
| | LightsState | LightStateEnum |
| | Marking | MarkingStruct |
| | PowerPlantOn | boolean |
| | RampDeployed | boolean |
| | SmokePlumePresent | boolean |
| | TentDeployed | boolean |
| | TrailState | TrailStateEnum |

| Cardinality | Units | Resolution |
|---|---|---|
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 0+ | N/A | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | TRUE/FALSE | N/A |
| 1 | N/A | N/A |

| Accuracy | Accuracy Condition | Update Type |
|---|---|---|
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Static |
| perfect | always | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| perfect | always | Conditional |
| N/A | N/A | Conditional |
| perfect | always | Conditional |
| perfect | always | Conditional |
| perfect | always | Static |
| perfect | always | Conditional |
| perfect | always | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| perfect | always | Conditional |
| perfect | always | Conditional |
| perfect | always | Static |
| perfect | always | Static |
| perfect | always | Static |
| N/A | N/A | Conditional |
| perfect | always | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Conditional |
| N/A | N/A | Static |
| perfect | always | Conditional |
| perfect | always | Conditional |
| perfect | always | Conditional |
| perfect | always | Conditional |
| N/A | N/A | Conditional |

| Update Condition | Transferable/Acceptable | Updateable/Reflectable |
|---|---|---|
| AccelerationChange | N | UR |
| AngVelocityChange | N | UR |
| On change | N | UR |
| On change | N | UR |
| N/A | N | UR |
| On change | N | UR |
| OrientationChange | N | UR |
| PositionChange | N | UR |
| VelocityChange | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| N/A | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| N/A | N | UR |
| N/A | N | UR |
| N/A | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| N/A | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |
| On change | N | UR |

· Attribute Table

| Routing Space |
|---|
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |
| N/A |

| Interaction | Parameter | Datatype |
|---|---|---|
| MunitionDetonation | ArticulatedPartsArray | ArticulatedParameterStruct |
| | DetonationLocation | PositionStruct |
| | DetonationResult | DetonationResultEnum |
| | EventID | EventIDStruct |
| | FiringObjectID [25] | RTIObjectIdStruct |
| | FinalVelocityVector | VelocityStruct |
| | FuseType | FuseTypeEnum |
| | MunitionObjectID [23] | RTIObjectIdStruct |
| | MunitionType | EntityTypeStruct |
| | QuantityFired | unsigned short |
| | RateOfFire | unsigned short |
| | RelativeDetonationLocation | RelativePositionStruct |
| | TargetObjectID [23] | RTIObjectIdStruct |
| | WarheadType | WarheadTypeEnum |
| WeaponFire | EventID | EventIDStruct |
| | FireControlSolutionRange | float |
| | FireMissionIndex | unsigned long |
| | FiringLocation | PositionStruct |
| | FiringObjectID [25] | RTIObjectIdStruct |
| | FuseType | FuseTypeEnum |
| | InitialVelocityVector | VelocityStruct |
| | MunitionObjectID [23] | RTIObjectIdStruct |
| | MunitionType | EntityTypeStruct |
| | QuantityFired | unsigned short |
| | RateOfFire | unsigned short |
| | TargetObjectID [23] | RTIObjectIdStruct |
| | WarheadType | WarheadTypeEnum |

| Cardinality | Units | Resolution |
|---|---|---|
| 0+ | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | metres | |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |
| 1 | N/A | N/A |

| Accuracy | Accuracy Condition | Routing Space |
|---|---|---|
| N/A | N/A | N/A |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| perfect | always | |
| perfect | always | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | N/A |
| perfect | always | |
| perfect | always | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| N/A | N/A | |
| perfect | always | |
| perfect | always | |
| N/A | N/A | |
| N/A | N/A | |

| Identifier | Enumerator |
|---|---|
| AcknowledgeFlagEnum | CreateEntity |
| | RemoveEntity |
| | StartResume |
| | StopFreeze |
| ActionEnum | Other |
| | LocalStorageOfTheRequestedInformation |
| | InformSimulationManagerOfRanOutOfAmmuniti |
| | InformSimulationManagerOfKilledInActionEvent |
| | InformSimulationManagerOfDamageEvent |
| | InformSimulationManagerOfMobilityDisabledEve |
| | InformSimulationManagerOfFireDisabledEvent |
| | InformSimulationManagerOfRanOutOfFuelEvent |
| | RecallCheckpointData |
| | RecallInitialParameters |
| | InitiateTetherLead |
| | InitiateTetherFollow |
| | Untether |
| | InitiateServiceStationResupply |
| | InitiateTailgateResupply |
| | InitiateHitchLead |
| | InitiateHitchFollow |
| | Unhitch |
| | Mount |
| | Dismount |
| | StartDailyReadinessCheck |
| | StopDailyReadinessCheck |
| | DataQuery |
| | StatusRequest |
| | SendObjectStateData |
| | Reconstitute |
| | LockSiteConfiguration |
| | UnlockSiteConfiguration |
| | UpdateSiteConfiguration |
| | QuerySiteConfiguration |
| | TetheringInformation |
| | MountIntent |
| | AcceptSubscription |
| | Unsubscribe |
| | TeleportEntity |
| | ChangeAggregateState |
| | RequestStartPDU |
| | WakeupGetReadyForInitialization |
| | InitializeInternalParameters |
| | SendPlanData |
| | SynchronizeInternalClocks |
| | Run |
| | SaveInternalParameters |
| | SimulateMalfunction |
| | JoinExercise |
| | ResignExercise |
| | TimeAdvance |

| Representation |
|---|
| 1 |
| 2 |
| 3 |
| 4 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |
| 16 |
| 17 |
| 18 |
| 19 |
| 20 |
| 21 |
| 22 |
| 23 |
| 24 |
| 25 |
| 26 |
| 27 |
| 28 |
| 29 |
| 30 |
| 31 |
| 33 |
| 34 |
| 35 |
| 36 |
| 37 |
| 38 |
| 39 |
| 40 |
| 41 |
| 42 |
| 43 |
| 44 |
| 45 |
| 46 |
| 47 |

| Identifier | Enumerator |
|---|---|
| ActionResultEnum | Other |
| | Pending |
| | Executing |
| | PartiallyComplete |
| | Complete |
| | RequestRejected |
| | RetransmitRequestNow |
| | RetransmitRequestLater |
| | InvalidTimeParameters |
| | SimulationTimeExceeded |
| | RequestDone |
| AggregateStateEnum | Other |
| | Aggregated |
| | Disaggregated |
| | FullyDisaggregated |
| | PseudoDisaggregated |
| | PartiallyDisaggregated |
| AntennaPatternEnum | OmniDirectional |
| | Beam |
| | SphericalHarmonic |
| ArticulatedTypeEnum | Other |
| | Rudder |
| | LeftFlap |
| | RightFlap |
| | LeftAileron |
| | RightAileron |
| | HelicopterMainRotor |
| | HelicopterTailRotor |
| | OtherAircraftControlSurfaces |
| | Periscope |
| | GenericAntenna |
| | Snorkel |
| | OtherExtendableParts |
| | LandingGear |
| | TailHook |
| | SpeedBrake |
| | LeftWeaponBayDoors |
| | RightWeaponBayDoors |
| | TankOrAPChatch |
| | Wingsweep |
| | BridgeLauncher |
| | BridgeSection1 |
| | BridgeSection2 |
| | BridgeSection3 |
| | PrimaryBlade1 |
| | PrimaryBlade2 |
| | PrimaryBoom |
| | PrimaryLauncherArm |
| | OtherFixedPositionParts |
| | PrimaryTurretNumber1 |
| | PrimaryTurretNumber2 |

| Representation |
|---|
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 0 |
| 1 |
| 2 |
| 0 |
| 1024 |
| 1056 |
| 1088 |
| 1120 |
| 1152 |
| 1184 |
| 1216 |
| 1248 |
| 2048 |
| 2080 |
| 2112 |
| 2144 |
| 3072 |
| 3104 |
| 3136 |
| 3168 |
| 3200 |
| 3232 |
| 3264 |
| 3296 |
| 3328 |
| 3360 |
| 3392 |
| 3424 |
| 3456 |
| 3488 |
| 3520 |
| 3552 |
| 4096 |
| 4128 |

| Identifier | Enumerator |
|---|---|
| | PrimaryTurretNumber3 |
| | PrimaryTurretNumber4 |
| | PrimaryTurretNumber5 |
| | PrimaryTurretNumber6 |
| | PrimaryTurretNumber7 |
| | PrimaryTurretNumber8 |
| | PrimaryTurretNumber9 |
| | PrimaryTurretNumber10 |
| | PrimaryGunNumber1 |
| | PrimaryGunNumber2 |
| | PrimaryGunNumber3 |
| | PrimaryGunNumber4 |
| | PrimaryGunNumber5 |
| | PrimaryGunNumber6 |
| | PrimaryGunNumber7 |
| | PrimaryGunNumber8 |
| | PrimaryGunNumber9 |
| | PrimaryGunNumber10 |
| | PrimaryLauncher1 |
| | PrimaryLauncher2 |
| | PrimaryLauncher3 |
| | PrimaryLauncher4 |
| | PrimaryLauncher5 |
| | PrimaryLauncher6 |
| | PrimaryLauncher7 |
| | PrimaryLauncher8 |
| | PrimaryLauncher9 |
| | PrimaryLauncher10 |
| | PrimaryDefenseSystems1 |
| | PrimaryDefenseSystems2 |
| | PrimaryDefenseSystems3 |
| | PrimaryDefenseSystems4 |
| | PrimaryDefenseSystems5 |
| | PrimaryDefenseSystems6 |
| | PrimaryDefenseSystems7 |
| | PrimaryDefenseSystems8 |
| | PrimaryDefenseSystems9 |
| | PrimaryDefenseSystems10 |
| | PrimaryRadar1 |
| | PrimaryRadar2 |
| | PrimaryRadar3 |
| | PrimaryRadar4 |
| | PrimaryRadar5 |
| | PrimaryRadar6 |
| | PrimaryRadar7 |
| | PrimaryRadar8 |
| | PrimaryRadar9 |
| | PrimaryRadar10 |
| | SecondaryTurretNumber1 |
| | SecondaryTurretNumber2 |
| | SecondaryTurretNumber3 |

| Representation |
|---|
| 4160 |
| 4192 |
| 4224 |
| 4256 |
| 4288 |
| 4320 |
| 4352 |
| 4384 |
| 4416 |
| 4448 |
| 4480 |
| 4512 |
| 4544 |
| 4576 |
| 4608 |
| 4640 |
| 4672 |
| 4704 |
| 4736 |
| 4768 |
| 4800 |
| 4832 |
| 4864 |
| 4896 |
| 4928 |
| 4960 |
| 4992 |
| 5024 |
| 5056 |
| 5088 |
| 5120 |
| 5152 |
| 5184 |
| 5216 |
| 5248 |
| 5280 |
| 5312 |
| 5344 |
| 5376 |
| 5408 |
| 5440 |
| 5472 |
| 5504 |
| 5536 |
| 5568 |
| 5600 |
| 5632 |
| 5664 |
| 5696 |
| 5728 |
| 5760 |

| Identifier | Enumerator |
|---|---|
| | SecondaryTurretNumber4 |
| | SecondaryTurretNumber5 |
| | SecondaryTurretNumber6 |
| | SecondaryTurretNumber7 |
| | SecondaryTurretNumber8 |
| | SecondaryTurretNumber9 |
| | SecondaryTurretNumber10 |
| | SecondaryGunNumber1 |
| | SecondaryGunNumber2 |
| | SecondaryGunNumber3 |
| | SecondaryGunNumber4 |
| | SecondaryGunNumber5 |
| | SecondaryGunNumber6 |
| | SecondaryGunNumber7 |
| | SecondaryGunNumber8 |
| | SecondaryGunNumber9 |
| | SecondaryGunNumber10 |
| | SecondaryLauncher1 |
| | SecondaryLauncher2 |
| | SecondaryLauncher3 |
| | SecondaryLauncher4 |
| | SecondaryLauncher5 |
| | SecondaryLauncher6 |
| | SecondaryLauncher7 |
| | SecondaryLauncher8 |
| | SecondaryLauncher9 |
| | SecondaryLauncher10 |
| | SecondaryDefenseSystems1 |
| | SecondaryDefenseSystems2 |
| | SecondaryDefenseSystems3 |
| | SecondaryDefenseSystems4 |
| | SecondaryDefenseSystems5 |
| | SecondaryDefenseSystems6 |
| | SecondaryDefenseSystems7 |
| | SecondaryDefenseSystems8 |
| | SecondaryDefenseSystems9 |
| | SecondaryDefenseSystems10 |
| | SecondaryRadar1 |
| | SecondaryRadar2 |
| | SecondaryRadar3 |
| | SecondaryRadar4 |
| | SecondaryRadar5 |
| | SecondaryRadar6 |
| | SecondaryRadar7 |
| | SecondaryRadar8 |
| | SecondaryRadar9 |
| | SecondaryRadar10 |
| BeamFunctionEnum | Other |
| | Search |
| | HeightFinder |
| | Acquisition |

| Representation |
|---|
| 5792 |
| 5824 |
| 5856 |
| 5888 |
| 5920 |
| 5952 |
| 5984 |
| 6016 |
| 6048 |
| 6080 |
| 6112 |
| 6144 |
| 6176 |
| 6208 |
| 6240 |
| 6272 |
| 6304 |
| 6336 |
| 6368 |
| 6400 |
| 6432 |
| 6464 |
| 6496 |
| 6528 |
| 6560 |
| 6592 |
| 6624 |
| 6656 |
| 6688 |
| 6720 |
| 6752 |
| 6784 |
| 6816 |
| 6848 |
| 6880 |
| 6912 |
| 6944 |
| 6976 |
| 7008 |
| 7040 |
| 7072 |
| 7104 |
| 7136 |
| 7168 |
| 7200 |
| 7232 |
| 7264 |
| 0 |
| 1 |
| 2 |
| 3 |

| Identifier | Enumerator |
|---|---|
| | Tracking |
| | AcquisitionAndTracking |
| | CommandGuidance |
| | Illumination |
| | RangeOnlyRadar |
| | MissileBeacon |
| | MissileFuze |
| | ActiveRadarMissileSeeker |
| | Jammer |
| CamouflageEnum | UniformPaintScheme |
| | DesertCamouflage |
| | WinterCamouflage |
| | ForestCamouflage |
| | GenericCamouflage |
| CharacterSetEnum | Other |
| | ASCII |
| | ArmyMarkingCCTT |
| | DigitChevron |
| CodeNameEnum | Other |
| | TBD |
| CollisionTypeEnum | Inelastic |
| | Elastic |
| CryptoSystemEnum | Other |
| | KY_28 |
| | KY_58 |
| | NarrowSpectrumSecureVoice_NSVE |
| | WideSpectrumSecureVoice_WSVE |
| | SINCGARS_ICOM |
| DamageStateEnum | NoDamage |
| | SlightDamage |
| | ModerateDamage |
| | Destroyed |
| DatumIDEnum | Dummy |
| DensityEnum | Clear |
| | Hazy |
| | Dense |
| | VeryDense |
| | Opaque |
| DesignatorCodeEnum | Other |
| | TBD |
| DetailedModulationEnum | Dummy |
| DetonationResultEnum | Other |
| | EntityImpact |
| | EntityProximateDetonation |
| | GroundImpact |
| | GroundProximateDetonation |
| | Detonation |
| | None |
| | HE_hit_Small |
| | HE_hit_Medium |
| | HE_hit_Large |

| Representation |
|---|
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 0 |
| 1 |
| 2 |
| 3 |
| 0 |
| 1 |
| 0 |
| 1 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 0 |
| 1 |
| 2 |
| 3 |
| 0 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 0 |
| 1 |
| 0 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |

| Identifier | Enumerator |
|---|---|
| | ArmorPiercingHit |
| | DirtBlast_Small |
| | DirtBlast_Medium |
| | DirtBlast_Large |
| | WaterBlast_Small |
| | WaterBlast_Medium |
| | WaterBlast_Large |
| | AirHit |
| | BuildingHit_Small |
| | BuildingHit_Medium |
| | BuildingHit_Large |
| | MineClearingLineCharge |
| | EnvironmentObjectImpact |
| | EnvironmentObjectProximateDetonation |
| | WaterImpact |
| | AirBurst |
| DRAlgorithmEnum | Other |
| | Static |
| | DRM_FPW |
| | DRM_RPW |
| | DRM_RVW |
| | DRM_FVW |
| | DRM_FPB |
| | DRM_RPB |
| | DRM_RVB |
| | DRM_FVB |
| EmitterFunctionEnum | Other |
| | MultiFunction |
| | EarlyWarningSurveillance |
| | HeightFinding |
| | FireControl |
| | AcquisitionDetection |
| | Tracking |
| | GuidanceIllumination |
| | FiringPointLaunchPointLocation |
| | Ranging |
| | RadarAltimeter |
| | Imaging |
| | MotionDetection |
| | Navigation |
| | Weather |
| | Instrumentation |
| | IdentificationClassification |
| | JammingNoise |
| | JammingDeception |
| | Decoy |
| | WeaponNonLethal |
| | WeaponLethal |
| EmitterNameEnum | Dummy |
| EntityCategoryEnum | Dummy |
| EntityCountryEnum | Dummy |

| Representation |
|---|
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |
| 16 |
| 17 |
| 18 |
| 19 |
| 20 |
| 21 |
| 22 |
| 23 |
| 24 |
| 25 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |
| 16 |
| 64 |
| 65 |
| 66 |
| 96 |
| 97 |
| 0 |
| 0 |
| 0 |

| Identifier | Enumerator |
|---|---|
| EntityDomainEnum | Dummy |
| EntityExtraEnum | Dummy |
| EntityKindEnum | Dummy |
| EntitySpecificEnum | Dummy |
| EntitySubcategoryEnum | Dummy |
| EventTypeEnum | Other |
| | Unused |
| | RanOutOfAmmunition |
| | KilledInAction |
| | Damage |
| | MobilityDisabled |
| | FireDisabled |
| | RanOutOfFuel |
| | EntityInitialization |
| | RequestForIndirectFireOrCASMission |
| | IndirectFireOrCASMission |
| | MinefieldEntry |
| | MinefieldDetonation |
| | VehicleMasterPowerOn |
| | VehicleMasterPowerOff |
| | AggregateStateChangeRequested |
| ForceIdEnum | Other |
| | Friendly |
| | Opposing |
| | Neutral |
| FormationEnum | Other |
| | Assembly |
| | Vee |
| | Wedge |
| | Line |
| | Column |
| FuseTypeEnum | Other |
| | IntelligentInfluence |
| | Sensor |
| | SelfDestruct |
| | UltraQuick |
| | Body |
| | DeepIntrusion |
| | Multifunction |
| | PointDetonation_PD |
| | BaseDetonation_BD |
| | Contact |
| | ContactInstantImpact |
| | ContactDelayed |
| | Contact10msDelay |
| | Contact20msDelay |
| | Contact50msDelay |
| | Contact60msDelay |
| | Contact100msDelay |
| | Contact125msDelay |
| | Contact250msDelay |

| Representation |
|---|
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 11 |
| 12 |
| 13 |
| 14 |
| 15 |
| 0 |
| 1 |
| 2 |
| 3 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 0 |
| 10 |
| 20 |
| 30 |
| 40 |
| 50 |
| 60 |
| 100 |
| 200 |
| 300 |
| 1000 |
| 1100 |
| 1200 |
| 1201 |
| 1202 |
| 1205 |
| 1206 |
| 1210 |
| 1212 |
| 1225 |

| Identifier | Enumerator |
|---|---|
| | ContactElectronicObliqueContact |
| | ContactGraze |
| | ContactCrush |
| | ContactHydrostatic |
| | ContactMechanical |
| | ContactChemical |
| | ContactPiezoelectric |
| | ContactPointInitiating |
| | ContactPointInitiatingBaseDetonating |
| | ContactBaseDetonating |
| | ContactBallisticCapAndBase |
| | ContactBase |
| | ContactNose |
| | ContactFittedInStandoffProbe |
| | ContactNonAligned |
| | Timed |
| | TimedProgrammable |
| | TimedBurnout |
| | TimedPyrotechnic |
| | TimedElectronic |
| | TimedBaseDelay |
| | TimedReinforcedNoseImpactDelay |
| | TimedShortDelayImpact |
| | Timed10msDelay |
| | Timed20msDelay |
| | Timed50msDelay |
| | Timed60msDelay |
| | Timed100msDelay |
| | Timed125msDelay |
| | Timed250msDelay |
| | TimedNoseMountedVariableDelay |
| | TimedLongDelaySide |
| | TimedSelectableDelay |
| | TimedImpact |
| | TimedSequence |
| | Proximity |
| | ProximityActiveLaser |
| | ProximityMagneticMagpolarity |
| | ProximityActiveDopplerRadar |
| | ProximityRadioFrequencyRF |
| | ProximityProgrammable |
| | ProximityProgrammablePrefragmented |
| | ProximityInfrared |
| | Command |
| | CommandElectronicRemotelySet |
| | Altitude |
| | AltitudeRadioAltimeter |
| | AltitudeAirBurst |
| | Depth |
| | Acoustic |
| | Pressure |

| Representation |
|---|
| 1300 |
| 1400 |
| 1500 |
| 1600 |
| 1700 |
| 1800 |
| 1900 |
| 1910 |
| 1920 |
| 1930 |
| 1940 |
| 1950 |
| 1960 |
| 1970 |
| 1980 |
| 2000 |
| 2100 |
| 2200 |
| 2300 |
| 2400 |
| 2500 |
| 2600 |
| 2700 |
| 2701 |
| 2702 |
| 2705 |
| 2706 |
| 2710 |
| 2712 |
| 2725 |
| 2800 |
| 2900 |
| 2910 |
| 2920 |
| 2930 |
| 3000 |
| 3100 |
| 3200 |
| 3300 |
| 3400 |
| 3500 |
| 3600 |
| 3700 |
| 4000 |
| 4100 |
| 5000 |
| 5100 |
| 5200 |
| 6000 |
| 7000 |
| 8000 |

| Identifier | Enumerator |
|---|---|
| | PressureDelay |
| | Inert |
| | Dummy |
| | Practice |
| | PlugRepresenting |
| | Training |
| | Pyrotechnic |
| | PyrotechnicDelay |
| | ElectroOptical |
| | ElectroMechanical |
| | ElectroMechanicalNose |
| | Strikerless |
| | StrikerlessNoseImpact |
| | StrikerlessCompressionIgnition |
| | CompressionIgnition |
| | CompressionIgnitionStrikerlessNoseImpact |
| | Percussion |
| | PercussionInstantaneous |
| | Electronic |
| | ElectronicInternallyMounted |
| | ElectronicRangeSetting |
| | ElectronicProgrammed |
| | Mechanical |
| | MechanicalNose |
| | MechanicalTail |
| HatchStateEnum | NotApplicable |
| | PrimaryHatchIsClosed |
| | PrimaryHatchIsPopped |
| | PrimaryHatchIsPoppedAndPersonIsVisibleUnde |
| | PrimaryHatchIsOpen |
| | PrimaryHatchIsOpenAndPersonIsVisible |
| InputSourceEnum | Other |
| | Pilot |
| | Copilot |
| | FirstOfficer |
| | Driver |
| | Loader |
| | Gunner |
| | Commander |
| | DigitalDataDevice |
| | Intercom |
| LifeformStateEnum | NotApplicable |
| | UprightStandingStill |
| | UprightWalking |
| | UprightRunning |
| | Kneeling |
| | Prone |
| | Crawling |
| | Swimming |
| | Parachuting |
| | Jumping |

| Representation |
|---|
| 8010 |
| 8100 |
| 8110 |
| 8120 |
| 8130 |
| 8150 |
| 9000 |
| 9010 |
| 9100 |
| 9110 |
| 9120 |
| 9200 |
| 9210 |
| 9220 |
| 9300 |
| 9310 |
| 9400 |
| 9410 |
| 9500 |
| 9510 |
| 9520 |
| 9530 |
| 9600 |
| 9610 |
| 9620 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |

| Identifier | Enumerator |
|---|---|
| | Sitting |
| | Squatting |
| | Crouching |
| | Wading |
| LightStateEnum | Other |
| MajorModulationEnum | Other |
| | Amplitude |
| | AmplitudeAndAngle |
| | Angle |
| | Combination |
| | Pulse |
| | Unmodulated |
| ModulationSystemEnum | Other |
| | Generic |
| | HQ |
| | HQII |
| | HQIIA |
| | SINCGARS |
| | CCTT_SINCGARS |
| MunitionTypeEnum | Other |
| NomenclatureEnum | Other |
| NomenclatureVersionEnum | Other |
| ParameterTypeEnum [9] | ArticulatedPart |
| | AttachedPart |
| ReceiveStateEnum | Off |
| | OnButNotReceiving |
| | OnAndReceiving |
| ReferenceSystemEnum [9] | WorldCoordinates |
| | EntityCoordinates |
| RepairResultEnum | AllRequestedRepairsPerformed |
| | NoRepairsPerformed |
| RepairTypeEnum | MotorOrEngine |
| | Starter |
| | Alternator |
| | Generator |
| | Battery |
| | EngineCoolantLeak |
| | FuelFilter |
| | TransmissionOilLeak |
| | EngineOilLeak |
| | Pumps |
| | Filters |
| | Transmission |
| | Brakes |
| | SuspensionSystem |
| | OilFilter |
| RequestStatusEnum | Other |
| | Pending |
| | Executing |
| | PartiallyComplete |
| | Complete |

| Representation |
|---|
| 10 |
| 11 |
| 12 |
| 13 |
| 0 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 0 |
| 0 |
| 0 |
| 0 |
| 1 |
| 0 |
| 1 |
| 2 |
| 1 |
| 2 |
| 0 |
| 1 |
| 10 |
| 20 |
| 30 |
| 40 |
| 50 |
| 60 |
| 70 |
| 80 |
| 90 |
| 100 |
| 110 |
| 120 |
| 130 |
| 140 |
| 150 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |

| Identifier | Enumerator |
|---|---|
| | RequestRejected |
| | RetransmitRequestNow |
| | RetransmitRequestLater |
| | InvalidTimeParameters |
| | SimulationTimeExceeded |
| | RequestDone |
| ResponseFlagEnum | Other |
| | AbleToComply |
| | UnableToComply |
| ServiceTypeEnum | Other |
| | Resupply |
| | Repair |
| StopFreezeReasonEnum | Other |
| | Recess |
| | Termination |
| | SystemFailure |
| | SecurityViolation |
| | EntityReconstitution |
| | StopForReset |
| | StopForRestart |
| | AbortTrainingResumeTacOps |
| TransmitStateEnum | Off |
| | OnButNotTransmitting |
| | OnAndTransmitting |
| TrailStateEnum | Other |
| WarheadTypeEnum | Other |
| | CargoVariableSubmunitions |
| | FuelAirExplosive |
| | GlassBeads |
| | Warhead_1um |
| | Warhead_5um |
| | Warhead_10um |
| | HighExplosive |
| | HE_Plastic |
| | HE_Incendiary |
| | HE_Fragmentation |
| | HE_Antitank |
| | HE_Bomblets |
| | HE_ShapedCharge |
| | HE_ContinuousRod |
| | HE_TungstenBall |
| | HE_BlastFragmentation |
| | HE_SteerableDartswithHE |
| | HE_Darts |
| | HE_Flechettes |
| | HE_DirectedFragmentation |
| | HE_SemiArmorPiercing |
| | HE_ShapedChargeFragmentation |
| | HE_SemiArmorPiercingFragmentation |
| | HE_HollowCharge |
| | HE_DoubleHollowCharge |

| Representation |
|---|
| 5 |
| 6 |
| 7 |
| 8 |
| 9 |
| 10 |
| 0 |
| 1 |
| 2 |
| 0 |
| 1 |
| 2 |
| 0 |
| 1 |
| 2 |
| 3 |
| 4 |
| 5 |
| 6 |
| 7 |
| 8 |
| 0 |
| 1 |
| 2 |
| 0 |
| 0 |
| 10 |
| 20 |
| 30 |
| 31 |
| 32 |
| 33 |
| 1000 |
| 1100 |
| 1200 |
| 1300 |
| 1400 |
| 1500 |
| 1600 |
| 1610 |
| 1615 |
| 1620 |
| 1625 |
| 1630 |
| 1635 |
| 1640 |
| 1645 |
| 1650 |
| 1655 |
| 1660 |
| 1665 |

| Identifier | Enumerator |
|---|---|
| | HE_GeneralPurpose |
| | HE_BlastPenetrator |
| | HE_RodPenetrator |
| | HE_Antipersonnel |
| | Smoke |
| | Illumination |
| | Practice |
| | Kinetic |
| | Mines |
| | Nuclear |
| | NuclearIMT |
| | ChemicalGeneral |
| | ChemicalBlisterAgent |
| | HD_Mustard |
| | ThickenedHD_Mustard |
| | DustyHD_Mustard |
| | ChemicalBloodAgent |
| | AC_HCN |
| | CK_CNCI |
| | CG_Phosgene |
| | ChemicalNerveAgent |
| | VX |
| | ThickenedVX |
| | DustyVX |
| | GA_Tabun |
| | ThickenedGA_Tabun |
| | DustyGA_Tabun |
| | GB_Sarin |
| | ThickenedGB_Sarin |
| | DustyGB_Sarin |
| | GD_Soman |
| | ThickenedGD_Soman |
| | DustyGD_Soman |
| | GF |
| | ThickenedGF |
| | DustyGF |
| | Biological |
| | BiologicalVirus |
| | BiologicalBacteria |
| | BiologicalRickettsia |
| | BiologicalGeneticallyModifiedMicroOrganisms |
| | BiologicalToxin |
| WeaponStateEnum | NoWeapon |
| | Stowed |
| | Deployed |
| | FiringPosition |
| AttributeChangeResultEnum | Dummy |
| CreateObjectResultEnum | Dummy |
| RemoveObjectResultEnum | Dummy |
| EncodingTypeEnum | Dummy |
| TacticalDataLinkTypeEnum | Dummy |

| Representation |
|---|
| 1670 |
| 1675 |
| 1680 |
| 1685 |
| 2000 |
| 3000 |
| 4000 |
| 5000 |
| 6000 |
| 7000 |
| 7010 |
| 8000 |
| 8100 |
| 8110 |
| 8115 |
| 8120 |
| 8200 |
| 8210 |
| 8215 |
| 8220 |
| 8300 |
| 8310 |
| 8315 |
| 8320 |
| 8325 |
| 8330 |
| 8335 |
| 8340 |
| 8345 |
| 8350 |
| 8355 |
| 8360 |
| 8365 |
| 8370 |
| 8375 |
| 8380 |
| 9000 |
| 9100 |
| 9200 |
| 9300 |
| 9400 |
| 9500 |
| 0 |
| 1 |
| 2 |
| 3 |
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |

| Identifier | Enumerator |
|---|---|
| UserProtocolEnum | Dummy |

| Representation |
|---|
| 0 |

Complex Datatype Table

| Complex Datatype | Field Name | Datatype | Cardinality | Units | Resolution |
|---|---|---|---|---|---|
| AccelerationStruct | XAcceleration | float | 1 | m/s/s | |
| | YAcceleration | float | 1 | m/s/s | |
| | ZAcceleration | float | 1 | m/s/s | |
| AngVelocityStruct | XAxisRate | float | 1 | radians/s | |
| | YAxisRate | float | 1 | radians/s | |
| | ZAxisRate | float | 1 | radians/s | |
| AntennaPatternStruct [7, 8] | BeamAntenna | BeamAntenna | 0-1 | N/A | N/A |
| | SphericalHarmoni | SphericalHar | 0-1 | N/A | N/A |
| ArticulatedParameterStruct | ParameterType | ParameterTyp | 1 | N/A | N/A |
| | ArticulatedParam | octet | 1 | N/A | 1 |
| | PartAttachedTo | unsigned short | 1 | N/A | 1 |
| | ArticulatedParam | ArticulatedTyp | 1 | N/A | N/A |
| | ParameterValue | ParameterVal | 1 | N/A | N/A |
| BeamAntennaStruct | BeamDirection | OrientationStr | 1 | N/A | N/A |
| | AzimuthBandwidt | float | 1 | radians | |
| | ElevationBeamwi | float | 1 | radians | |
| | ReferenceSystem | ReferenceSyst | 1 | N/A | N/A |
| | Padding1 | octet | 1 | N/A | 1 |
| | Padding2 | short | 1 | N/A | 1 |
| | Ez | float | 1 | | |
| | Ex | float | 1 | | |
| | Phase | float | 1 | | |
| ClockTimeStruct | Hours | long | 1 | hours | 1 |
| | TimePastTheHou | unsigned long | 1 | 1.626 microse | 1.626 microse |
| DimensionStruct | XAxisLength | float | 1 | metres | |
| | YAxisLength | float | 1 | metres | |
| | ZAxisLength | float | 1 | metres | |
| EntityTypeStruct [6] | EntityKind | EntityKindEnu | 1 | N/A | N/A |
| | Domain | EntityDomainE | 1 | N/A | N/A |
| | Country | EntityCountry | 1 | N/A | N/A |
| | Category | EntityCategory | 1 | N/A | N/A |
| | Subcategory | EntitySubcate | 1 | N/A | N/A |
| | Specific | EntitySpecific | 1 | N/A | N/A |
| | Extra | EntityExtraEnu | 1 | N/A | N/A |
| EntityIDStruct | FederateID | FederateIDStr | 1 | N/A | N/A |
| | EntityNumber | unsigned short | 1 | N/A | 1 |
| EventIDStruct | EventCount | unsigned short | 1 | N/A | N/A |
| | IssuingObjectID | RTIObjectIdSt | 1 | N/A | N/A |
| FederateIDStruct | SiteID | unsigned short | 1 | N/A | N/A |
| | ApplicationID | unsigned short | 1 | N/A | N/A |
| FixedDatumStruct | FixedDatumID | DatumIDEnum | 1 | N/A | N/A |
| | FixedDatumValue | unsigned long | 1 | | |
| MarkingStruct | CharacterSet | CharacterSetE | 1 | N/A | N/A |
| | MarkingString | octet | 11 | _ [2] | |
| ModulationStruct | DummyModulatio | any | 1 | | |
| OrientationStruct | Psi | float | 1 | radians | |
| | Theta | float | 1 | radians | |
| | Phi | float | 1 | radians | |
| ParameterValueT | Dummy | any | 1 | | |

| Accuracy | Accuracy Condition |
|---|---|
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| N/A | N/A |
| perfect | always |
| perfect | always |
| N/A | N/A |
| perfect | always |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |

| Complex Datatype | Field Name | Datatype | Cardinality | Units | Resolution |
|---|---|---|---|---|---|
| PositionStruct | X | double | 1 | metres | |
| | Y | double | 1 | metres | |
| | Z | double | 1 | metres | |
| RadioTypeStruct | EntityKind | EntityKindEnu | 1 | N/A | N/A |
| | Domain | EntityDomainE | 1 | N/A | N/A |
| | Country | EntityCountry | 1 | N/A | N/A |
| | Category | EntityCategory | 1 | N/A | N/A |
| | Subcategory | EntitySubcate | 1 | N/A | N/A |
| | NomenclatureVer | Nomenclature | 1 | N/A | N/A |
| | Nomenclature | Nomenclature | 1 | N/A | N/A |
| RelativePositionStruct | BodyX | float | 1 | metres | |
| | BodyY | float | 1 | metres | |
| | BodyZ | float | 1 | metres | |
| RTIObjectIdArrayStruct [5] | Length | unsigned short | 1 | N/A | N/A |
| | ID | string | 1 | N/A | N/A |
| RTIObjectIdStruc | ID | string | 1 | N/A | N/A |
| SilentAggregateStruct | AggregateType | EntityTypeStru | 1 | N/A | N/A |
| | NumberOfAggreg | unsigned short | 1 | N/A | 1 |
| SilentEntityStruct | EntityType | EntityTypeStru | 1 | N/A | N/A |
| | NumberOfEntities | unsigned short | 1 | N/A | 1 |
| SphericalHarmonicAntennaStruct [12] | Order | char | 1 | N/A | 1 |
| | Coefficients | float | 1+ | | |
| | ReferenceSystem | ReferenceSyst | 1 | N/A | N/A |
| SupplyStruct | SupplyType | EntityTypeStru | 1 | N/A | N/A |
| | Quantity | float | 1 | N/A | N/A |
| UnsignedInteger6 | Dummy64 | any | 1 | | |
| VariableDatumSetStruct | NumberOfVariabl | unsigned long | 1 | N/A | 1 |
| | VariableDatums | VariableDatum | 1+ | N/A | N/A |
| VariableDatumStruct | DatumID | DatumIDEnum | 1 | N/A | N/A |
| | DatumLength | unsigned long | 1 | N/A | 1 |
| | DatumValue | any | 1 | | |
| VelocityStruct | XVelocity | float | 1 | m/s | |
| | YVelocity | float | 1 | m/s | |
| | ZVelocity | float | 1 | m/s | |
| AttributeValueSet | AttributeSetCount | any | 1 | | |

| Accuracy | Accuracy Condition |
|---|---|
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| perfect | always |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| N/A | N/A |
| N/A | N/A |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |
| perfect | always |

| Term | Definition |
|---|---|
| BaseEntity | A base class of all scenario domain participants, both aggregate and discrete. The Ba |
| CivilAirLandPlatform | A civilian platform entity that operates mainly in the air, such as aircraft, balloons, etc. |
| CivilAmphibiousPlatform | A civilian platform entity that can operate both on the land and the sea. |
| Civilian | A civilian (human). |
| CivilLandPlatform | A civilian platform entity that operates wholy on the surface of the earth. |
| CivilMultiDomainPlatform | A civilian platform entity that operates in more than one domain (excluding those comb |
| CivilPlatform | A civilan platform entity. |
| CivilSeaSurfacePlatform | A civilian platform entity that operates wholy on the surface of the sea. |
| CivilSpacePlatform | A civilian platform entity that operates mainly in space. |
| CivilSubmersiblePlatform | A civilian platform entity that operates either on the surface of the sea, or beneath it. |
| MilitaryAirLandPlatform | A military platform entity that operates mainly in the air, such as aircraft, balloons, etc. |
| MilitaryAmphibiousPlatform | A military platform entity that can operate both on the land and the sea. |
| MilitaryEntity | An object which has position and fixed size and shape and is under the control of arm |
| MilitaryLandPlatform | A military platform entity that operates wholy on the surface of the earth. |
| MilitaryMultiDomainPlatform | A military platform entity that operates in more than one domain (excluding those com |
| MilitaryPlatformEntity | A physical object under the control of armed forces upon which sensor, communicatio |
| MilitarySeaSurfacePlatform | A military platform entity that operates wholy on the surface of the sea. |
| MilitarySpacePlatform | A military platform entity that operates mainly in space. |
| MilitarySubmersiblePlatform | A military platform entity that operates either on the surface of the sea, or beneath it. |
| MunitionEntity | A complete device charged with explosives, propellants, pyrotechnics, initiating compo |
| PhysicalEntity | A base class of all discrete platform scenario domain participants. |
| Soldier | A human who is a member of an armed force. |

| Term | Definition |
|---|---|
| MunitionDetonation | Communicates information associated with the impact or detonation of a munition |
| WeaponFire | Communicates information associated with the firing or launch of a munition. |

| Class | Term | Definition |
|---|---|---|
| BaseEntity | AccelerationVector | The magnitude of the change in linear velocity of an object over time. |
| | AngularVelocityVect | The rate at which an entity's orientation is changing over time. |
| | DRAlgorithm | Dead reckoning algorithm used by the issuing object. |
| | EntityType | The category of the entity. |
| | EntityID | The unique identifier for the entity instance. |
| | IsFrozen | Whether the entity is frozen or not. |
| | Orientation | The yaw, pitch and roll angles between the entity's attitude and the refer |
| | Position | Location of the entity. |
| | VelocityVector | The rate at which an entity's position is changing over time |
| MilitaryEntity | AlternateEntityType | The category of entity to be used when viewed by entities on the "oppos |
| | CamouflageType | The type of camouflage in use (if any). |
| | FirePowerDisabled | Whether the entity's main weapon system has been disabled or not. |
| | ForceID | The identification of the force that the entity belongs to. |
| | IsConcealed | Whether the entity is concealed or not. |
| MilitaryPlatformEntit y | AfterburnerOn | Whether the entity's afterburner is on or not. |
| | HasAmmunitionSup | Whether the entity has the capability to supply other entities with ammu |
| | LauncherRaised | Whether the entity's weapon launcher is in the raised position. |
| MunitionEntity | LauncherFlashPrese | Whether the flash of the munition being launched is present or not. |
| PhysicalEntity | ArticulatedParamete | Identification of the visible parts, and their states, of the entity which are |
| | DamageState | The state of damage of the entity. |
| | EngineSmokeOn | Whether the entity's engine is generating smoke or not. |
| | FlamesPresent | Whether the entity is on fire (with visible flames) or not. |
| | HasFuelSupplyCap | Whether the entity has the capability to supply other entities with fuel or |
| | HasRecoveryCap | Whether the entity has the capability to recover other entities or not. |
| | HasRepairCap | Whether the entity has the capability to repair other entities or not. |
| | HatchState | The state of the entity's (main) hatch. |
| | Immobilized | Whether the entity is immobilized or not. |
| | LifeformState | The state of the lifeform (if the entity is a lifeform). |
| | LightsState | The state of the entity's lights |
| | Marking | A unique marking or combination of characters used to distinguish the e |
| | PowerPlantOn | Whether the entity's power plant is on or not. |
| | RampDeployed | Whether the entity has deployed a ramp or not. |
| | SmokePlumePresen | Whether the entity is generating smoke or not (intentional or unintention |
| | TentDeployed | Whether the entity has deployed tent or not. |
| | TrailState | The type and size of any trail that the entity is making. |

| Interaction | Term | Definition |
|---|---|---|
| MunitionDetonation | ArticulatedPartsArra | The set of articulated parts affected by the detonation (including where |
| | DetonationLocation | The location, in the world coordinate system, of the detonation. |
| | DetonationResult | The type of detonation (including no detonation). |
| | EventID | An ID, generated by the issuing federate, used to associated related fire |
| | FiringObjectID | The ID of the object firing the munition. |
| | FinalVelocityVector | The velocity vector of the munition at the moment of the detonation. |
| | FuseType | The type of fuse on the munition. |
| | MunitionObjectID | The ID of the associated munition object (if any). |
| | MunitionType | The type of munition that is detonating. |
| | QuantityFired | The quantity of rounds fired in a burst. |
| | RateOfFire | The rate of fire, in rounds per minute, of the munitions in the burst. |
| | RelativeDetonationL | The location of the detonation, relative to the target object (if any). |
| | TargetObjectID | The ID of the object that the munition has detonated on. |
| | WarheadType | The type of warhead on the munition. |
| WeaponFire | EventID | An ID, generated by the issuing federate, used to associated related fire |
| | FireControlSolutionR | The range used in the fire control solution. Zero if the range is unknown |
| | FireMissionIndex | A unique index to identify the fire mission (used to associated weapon fi |
| | FiringLocation | The location, the world coordinate system, of the weapon fire. |
| | FiringObjectID | The ID of the object ffiring the munition. |
| | FuseType | The type of fuse on the munition. |
| | InitialVelocityVector | The velocity vector of the munition when fired. |
| | MunitionObjectID | The ID of the associated munition object (if any). |
| | MunitionType | The type of munition being fired. |
| | QuantityFired | The number of rounds fired in the fire event. |
| | RateOfFire | The rate of fire at which the munitions in the burst described in the fire e |
| | TargetObjectID | The ID of the object being fired at (if any). |
| | WarheadType | The type of warhead fitted to the munition being fired. |

Notes

| ID | Text |
|---|---|
| 1 | The choice of whether an entity is a physical entity rather than an environmental entity, boils down to |
| 10 | This is a 16-bit enumeration |
| 11 | This is a 32-bit enumeration |
| 12 | This structure is taken directly from the IEEE 1278.1-1995 (DIS) definition of the Spherical Harmonic |
| 13 | The Acknowledge interaction is issued in response to the CreateEntity, RemoveEntity, StartResume, |
| 14 | This is a timestamp record (see DIS 5.2.31) |
| 15 | Or any attribute or private data identified by a DatumID enumeration. |
| 16 | This comment has been deleted! |
| 17 | IEEE 1278.1-1995 specifies that the comment PDU (upon which the Comment interaction is based) |
| 18 | If the EntityNumber field is set to RQST_ASSIGN_ID (hex FFFE) then the receiving application shou |
| 19 | The Request ID is a monotonically increasing integer identifier inserted by the Simulation Manager in |
| 2 | The units for the MarkingString are specified by the value of the CharacterSet. |
| 20 | This field matches this response with the specific ActionRequest interaction sent by the simulation m |
| 21 | This field matches this response with the specific SetData or DataQuery interaction sent by the simul |
| 22 | This field matches this response with the specific RemoveObject interaction sent by the simulation m |
| 23 | If there is no object instance associated with the attribute, then this should be set to the empty string |
| 24 | If there are no objects to be referenced in the array then the count should be set to zero, and the ID s |
| 25 | This must reference a valid Object instance. |
| 3 | The AttributeChangeResult interaction should be sent in response to an AttributeChangeRequest int |
| 4 | This is the unique ObjectName associated with each object instance. The user can define the name |
| 5 | This is a series of ObjectNames, the number of ObjectNames is stored in the Count field, with the str |
| 6 | Currently the entity type record is a variant record as in DIS (the meaning of the enumerations in eac |
| 7 | The current OMT standard does not allow the definition of variant records in complex data types, i.e. |
| 8 | The contents of the AntennaPatternStruct complex datatype depends on the value of the AntennaPa |
| 9 | This is an 8-bit enumeration |

# APPENDIX I

# INSTALLATION TRIP REPORT

# MEMO

To: HLA SBIR TEAM
From: Randy Lang
Date: 12/17/99
Subject: Delivery of SBIR Topic N96-053 Phase II

This is a summary of significant events and facts concerning the delivery of the N96-053 SBIR.

Randy Lang and Willie Bush arrived at NAS Oceanna Tuesday 12/14/99. Ray Shaw was our point of contact.

COMSEC at oceana (Chief Kitchens) issued 3 secure data modems and keys to Randy Lang to support testing.

All equipment bought under the SBIR was transferred to Ray Shaw with the exception of a Dell computer system that was transferred to Jeff Davis. There is one Intergraph computer System that was not delivered (It is being repaired and will be shipped to Ray Shaw upon completion of repairs).

All software and documentation was inventoried and delivered to Ray Shaw.

The HLA network and associated demo was shown to Ray Shaw and Jeff Davis. The HLA equipment is setup in the Debrief room in building 150.

An analog phone line was run by Dual from the computer room up to the IOS station at the TES. Our design required two lines but we were able test each task independently using 1 line.

A secure data connection was accomplished from building 150 to the F14D trainer. User Instructions and training was supplied to Jeff Davis. Online information is also provided. Two SECTEL model 1500 secure data modems were used for this connection.

The secure voice interface equipment was installed in cabinet 41. Telephone to intercom interface was tested unsuccessfully. The cause was undetermined after troubleshooting of some government and Dual equipment.

The secure Data devices model 1910 were unable to establish connections. The 1500's sufficed for the purposes of the demo. All 3 secure data modems were returned to Chief Kitchens Friday 12/17/99.

Ray Shaw was briefed Friday Morning of our status prior to our departure.

A final report will be delivered as a conclusion to this effort. This report will include: Software documentation and design for all Dual developed software/ hardware; guides for using equipment; and project conclusions/ lessons learned.